LATHAM & WATKINS LLP
Douglas E. Lumish (SBN 183863)
doug.lumish@lw.com
Gabriel S. Gross (SBN 254672)
gabe.gross@lw.com
Arman Zahoory (SBN 306421)
arman.zahoory@lw.com
Rachel S. Horn (SBN 335737)
rachel.horn@lw.com
140 Scott Drive
Menlo Park, California 94025
Telephone: (650) 328-4600
Facsimile: (650) 463-2600

Attorneys for Defendant and Counterclaimant
Skyryse, Inc.

UNITED STATES DISTRICT COURT

CENTRAL DISTRICT OF CALIFORNIA

MOOG INC.,

Plaintiff,

v

SKYRYSE, INC., ROBERT ALIN PILKINGTON, MISOOK KIM, and DOES NOS. 1-50,

Defendants.

SKYRYSE, INC.,

Counterclaimant,

v

MOOG INC.,

Counterclaim-Defendant.

CASE NO. 2:22-cv-09094-GW-MAR

DECLARATION OF MICHAEL J. DREIKORN IN SUPPORT OF DEFENDANT AND COUNTERCLAIMANT SKYRYSE, INC.'S OPPOSITION TO MOOG'S MOTION TO ENFORCE COMPLIANCE WITH MARCH 11, 2022 STIPULATED TRO AND FOR MONETARY AND ADVERSE INFERENCE SANCTIONS

Judge: Hon. George H. Wu
Crtrm: 9D

REDACTED VERSION OF DOCUMENT PROPOSED TO BE FILED UNDER SEAL

DECLARATION OF MICHAEL J. DREIKORN, Ed.D.

I, Michael J. Dreikorn, declare as follows:

I. INTRODUCTION AND SUMMARY OF OPINIONS

1. I have been retained by the law firm Latham & Watkins, LLP ("Latham"), on behalf of Skyryse, Inc. ("Skyryse"). I provide this declaration in support of Skyryse's Opposition to Moog Inc.'s ("Moog") Motion to Enforce Compliance with the March 11, 2022 Stipulated TRO ("Motion").

2. I was asked to respond to certain opinions of Kevin Crozier and Bruce Pixley, who provided expert declarations dated 16 March, 2023 in support of Moog's Motion ("Crozier Decl." and "Pixley Decl."). Specifically, I was asked to assess plans, checklists, and standards documents and templates associated with flight certification, including related to the software certification process, and to opine as to their consistency with FAA certification requirements and industry standards, including to the extent such documents are similar between Moog and Skyryse, and to respond to Mr. Pixley and Mr. Crozier's opinions that the documents identified in their declarations constitute Moog non-public Information.

3. Mr. Crozier opines that certain data produced in this case "contains substantial evidence that Skyryse personnel (using Skyryse e-mail accounts and devices) were possessing, disclosing to third-parties, accessing, and using Moog non-public information," including "Moog software process documents and checklists."[1] Mr. Pixley also opines that he believes to have "[f]ound many examples of Moog non-public data that was transmitted by Skyryse email accounts," which he claims to have found in documents produced by third parties such as Rex Hyde and Hummingbird.[2]

[1] Crozier Decl. ¶¶ 11-14.

[2] *See*, *e.g.*, Pixley Decl. ¶¶ 22, 34, 40.

4. I have reviewed these process documents, including software process documents and checklists, that Mr. Crozier and Mr. Pixley identify in their declarations, and I disagree with their conclusions that these documents reflect that "Skyryse personnel were possessing, disclosing to third parties, accessing, and using Moog non-public information."[3] This is because, as I explain below, I am aware of facts showing that the contents of such documents are based on, and often copied from, publicly available industry standards that are used by many companies in the industry seeking certification of airborne software other than Moog.

5. Specifically, Mr. Pixley and Mr. Crozier focus on six types of documents that they claim constitute Moog non-public information: (1) DPA and Software Process Checklists; (2) Plan for Software Aspects of Certification ("PSAC"); (3) Software Quality Assurance Plan ("SQAP"); (4) Software Configuration Management Plan ("SCMP"); (5) Software Development Plan ("SDP"); and (6) JIRA and SVN guides. Mr. Pixley and Mr. Crozier point to the fact that certain Skyryse contractors and personnel were exchanging these documents, which they opine constitute Moog non-public information.

6. I understand from reviewing their deposition transcripts that [redacted]

[3] *See, e.g.*, Crozier Decl. ¶¶¶ 11-15.

[redacted]. Both experts, however, opine that these documents constitute Moog non-public information, and their existence constitutes "evidence of misappropriation" by Skyryse and its employees.[6] But without having investigated the origins of the underlying documents and the content contained in them, I am not aware of any factual support for these opinions.

7. The Moog and Skyryse software process checklists Mr. Crozier and Mr. Pixley reference in their declarations[7] are typical of checklists I have frequently seen in my experience in the aviation industry. These types of checklists are widely used in industry, and their form, structure, and content are standardized. In general, the information contained in these checklists is not unique to Moog and is the type of information that is known to the public, and often copied verbatim from publicly available industry standards that are not unique to Moog but are used by all companies in the field.[8] Nothing identified by Mr. Crozier or Mr. Pixley as "evidence of Moog non-public information" suggests otherwise.

8. The Moog and Skyryse airborne software certification process documents Mr. Crozier references are typical of software documentation I have

[4] *See, e.g.*, [redacted]

[5] *See, e.g.*, [redacted]

[6] *See, e.g.*, Pixley Decl. ¶¶ 25-28, 30, 35-36, 40; Crozier Decl. ¶¶ 11, 20-23, 40-42.

[7] *See, e.g.*, Pixley Decl. ¶¶ 25-28, 30, 35-36, 40; Crozier Decl. ¶¶ 11, 20-23, 40-42.

[8] *See infra* § IV.B. The relevant standards include the DO-178C and DO-330 standards, and similar checklists are available for purchase from third-party companies.

frequently seen in my experience in the aviation industry. These types of software process documents are widely used in industry, and their form, structure, and content are standardized, and often copied verbatim from publicly available industry standards. In general, the information contained in these documents is not unique to Moog and is the type of information that is known to the public.[9] Nothing identified by Mr. Crozier or Mr. Pixley as what he calls "evidence of Moog non-public information" suggests otherwise.

9. Moog's experts also opine that Skyryse's airborne software certification procedures and planning are "nearly identical" to that of Moog. Where similarities in wording and structure exist, it is standards-driven[10] and not a product of any unique efforts by Moog. Moog's experts do not provide any discussion of industry standardization and the public availability of many of the topics for which they criticize Skyryse.

II. QUALIFICATIONS

10. I have been professionally engaged in the aviation industry for over 43 years and am considered an expert in the field of regulatory compliance, aviation product design, manufacturing, quality, supply-chain management, government contracts, and maintenance.

11. Since 2002, I have owned and managed The IPL Group, LLC, providing services to governmental entities and private companies in the aviation, space and defense (AS&D) industries, for aircraft certification, engineering, production certification, regulatory compliance, quality assurance, maintenance activities, human factors, organizational performance, and other AS&D matters. Many of our engineering projects include the installation of airborne software that requires FAA or other civil aviation authority certification. Since 2004, our firm's expert witness services have been provided through ASD Experts, LLC.

[9] For example, in the publicly available DO-178 and DO-330 standards.

[10] Ex. D2, DO-178C; Ex. D3, DO-330, and Ex. D4, DO-331.

12. Prior to forming The IPL Group, I was the Vice President of Quality and Regulatory Compliance for Pratt & Whitney, an engine manufacturer. I was responsible for the quality, product integrity, and regulatory compliance of a global production (including suppliers) and design system that held approvals from the FAA, TCCA, EASA, Defense Contract Management Agency (DCMA), National Aeronautics and Space Administration (NASA), and others. The products of Pratt & Whitney also included embedded airborne software that required FAA, DoD, NASA, or other civil aviation authority certification.

13. Prior to Pratt & Whitney, I was the Assistant Division Manager (AIR-201) of the Federal Aviation Administration's (FAA's) Aircraft Certification, Production and Airworthiness division, located in Washington, D.C. In this role, my responsibilities included the development of national policy (including the use and international harmonization of FAA Form 8130-3, industry acceptance of AS9100, and development of national policy related to airborne software), FAA regulations, FAA advisory materials, harmonization with ICAO rules; service-wide human resource staffing; and the regulatory compliance and enforcement of the U.S. civil aviation manufacturing system. I was also instrumental in the development of implementation procedures for bilateral aviation safety agreements between the United States and various partner countries.

14. Other professional and management positions included McDonnell Douglas, Northrop, and the U.S. Army.

15. Throughout my professional career, I have maintained various professional certifications (including FAA airframe and powerplant mechanic, FAA inspection authorization, FAA designated airworthiness representative, journeyman helicopter mechanic, and aerospace lead auditor), performed numerous aviation-related investigations, led audits of aviation organizations,

published books[11] and journal articles related to aviation quality, certification, and maintenance systems, and actively engaged in the development of industry standards related to quality, product verification methods, maintenance practices, and human factors.

16. I am a fellow of the American Society for Quality (ASQ), senior member of the American Institute for Aeronautics and Astronautics (AIAA), past-chair for the Aviation, Space & Defense division of ASQ, founding member of the International Aerospace Quality Group (IAQG),[12] founder of the Aerospace and Defense Learning Institute (ADLI),[13] and a member of various other aerospace related professional societies.

17. I hold a doctorate in the field of Human Resource and Organizational Development from the George Washington University, a masters in Management from Friends University, and a bachelors in Professional Aeronautics from Embry-Riddle Aeronautical University. A more complete description of my professional background is set forth in Exhibit D1.

III. BACKGROUND

18. As background for my opinions, I provide the following information on the aviation industry, certification requirements for airborne software on aeronautical systems, and industry standards for airborne software development and documentation.

A. Certification Requirements for Airborne Software

19. Software that is embedded into the components and systems of an aircraft is referred to as "airborne software" and is subject to regulatory certification for use in civil aviation and to Department of Defense (DoD)

[11] Published books include, Aviation Industry Quality Systems: ISO9000 and the Federal Aviation Regulations (1995); The Synergy of One: Creating High-Performing Sustainable organizations through integrated performance leadership (2003); The standard of knowledge for the aviation, space & defense industry practitioner (2012).

[12] The IAQG is the organization responsible for the development of AS9100-series quality standards.

[13] The ADLI is the developer and custodian of the aviation, space and defense (AS&D) industry quality management body of knowledge of AS&D quality professionals.

certification for military aircraft. For civil aviation applications, airborne software is subject to the regulatory certification requirements of the Federal Aviation Administration (FAA) and other national civil aviation authorities such as European Aviation Safety Agency (EASA) and Transport Canada. As all relevant civil aviation authorities are signatories (through their national Departments of State) to the Chicago Convention and subscribe to the charter of the International Civil Aviation Organization (ICAO), the design and product certification regulatory requirements for airborne software are similar globally.

20. Governmental aviation regulations focus on the end-requirement for safety and point to industry standards as an acceptable means of compliance. The FAA prescribes design certification standards for aircraft, engines, and products through 14 CFR Chapter I, Subchapter C of the Federal Aviation Regulations (FAR). In addition to a standalone Part for aircraft engines, Subchapter C provides design safety standards for each major category of aircraft. For example:

- Part 21 provides the overall "Certification Procedures for Products and Articles";
- Part 23 provides the specific airworthiness standards for "Normal Category Airplanes";
- Part 25 provides the specific airworthiness standards for "Transport Category Airplanes";
- Part 27 provides the specific airworthiness standards for "Normal Category Rotorcraft";
- Part 29 provides the specific airworthiness standards for "Transport Category Rotorcraft"; and
- Part 33 provides the specific airworthiness standards for "Aircraft Engines."

21. Each of the above noted FAR Parts contain sections that relate to "System Safety" and require an "Applicant" to design and install systems so that "(a) Each catastrophic failure condition – (1) Is extremely improbable; and

(2) Does not result from a single failure. (b) Each hazardous failure condition is extremely remote."[14]

22. While the specific FAR does not prescribe how to comply with the system safety requirements, the FAA provides guidance through its Advisory Circulars (AC). ACs relevant to each of the aircraft categories[15] point to AC 20-115D as a means of compliance. AC 20-115D refers to AC 20-152A and FAA Orders 8110.49 and 8110.105A as an acceptable means for showing compliance with the pertinent airworthiness requirements (design safety regulations).

23. Both AC 20-115D and 20-152A provide a description of "an acceptable means, but not the only means, for showing compliance with the applicable airworthiness regulations for the software aspects of airborne systems and equipment in type certification or TSO authorization. This AC is not mandatory and does not constitute a regulation. However, if you use the means described in the AC, you must follow it in all applicable respects." And, both ACs direct to an industry standard authored by the RTCA[16] referred to as DO-178C. The method for showing compliance for airborne software is the same for all categories of aircraft and is based on the standards contained in the DO-178C.

24. The FAA directs its technical community (inspectors, engineers, and designees) on the processes and forms for design certification through a series of FAA Orders. FAA Order 8110.49 provides guidelines for "the approval of airborne systems and equipment and the software aspects of those systems related to type certificates (TC), supplemental type certificates (STC), amended type certificates (ATC), amended supplemental type certificates (ASTC), and technical standard order (TSO) authorizations (TSOA)." The Order specifically directs

[14] 14 CFR 25.1709.

[15] AC 23.1309-1E; 25.1309-1A; 27-1B; 29-2C; and 33.75-1A.

[16] Note: RTCA was previously known as the Radio Technical Commission for Aeronautics.

guidance to DO-178C and prescribes a standardized certification methodology and documentation structure.

25. With regard to the interaction between the applicant and the certification authority (FAA), FAA Order 8110.49 directs to DO-178C Section 9. Section 9 states "[t]he objectives of the certification liaison process are to: a. Establish communication and understanding between the applicant and the certification authority throughout the software life cycle to assist the certification process.; b. Gain agreement on the means of compliance through approval of the Plan for Software Aspects of Certification.; c. Provide compliance substantiation."

26. FAA Order 8110.49 further defines the structure and content for certification of airborne software. In Chapter 2.2.b. of the Order, the FAA provides "… the certification authority should include the following practical arrangements with the software developer:

> (1) Agreement on the scope of review(s) that will be conducted.
> (2) Agreement on date(s) and location(s) of the review(s).
> (3) Identification of the certification authority's personnel involved.
> (4) Identification of any designees involved.
> (5) Development of the agenda(s) and expectations.
> (6) Listing of software data to be made available (both before and at the review(s)).
> (7) Clarification of procedures to be used.
> (8) Identification of any required resources.
> (9) Specification of date(s) and means for communicating review results (may include corrective actions and other post-review activities)."

27. FAA Order 8110.105A provides guidance for approving both simple and complex custom micro-coded components and directs the FAA technical community to DO-254. The "guidance applies to airborne systems and equipment, and the airborne electronic hardware of those systems when you work in a certification project (i.e., type, supplemental, amended, and amended supplemental) or technical standard order authorization." While FAA Order

8110.105A focuses on micro-coded components and not specifically on airborne software, it does direct to DO-178C if any embedded logic is to be modified in such components for the processes of certification.

28. The FAA does not certify airborne software independent of its installation on a certifiable aircraft, product or appliance. Airborne software is FAA-approved only when incorporated as part of a stand-alone product or appliance, such as those previously described in this declaration or as part of a Technical Standard Order Authorization (TSOA).[17] Through FAA Order 8110.4C the FAA directs its certification personnel to process applications for FAA design certification within a very defined and structured model.[18] (Reference Figure 1.)

[17] 14 CFR 21 Subpart O for the TSOA regulations.

[18] FAA Order 8110.4C, Chapter 2.2.

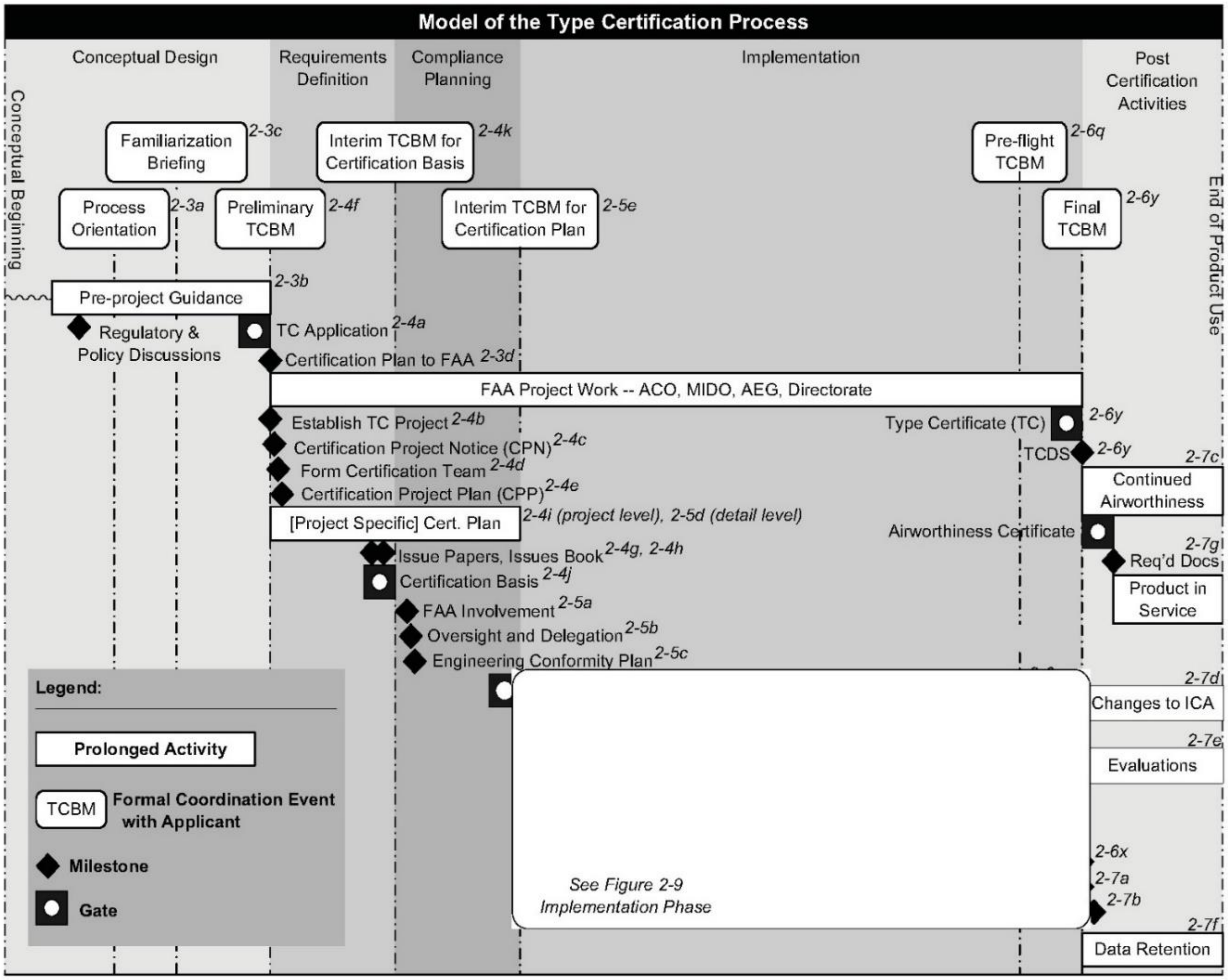


Figure 1., The Typical FAA Type Certification Process.[19]

29. As provided in FAA Order 8110.4C, the FAA design certification (type certification) process follows a lifecycle model and includes the phases of conceptual design, requirements definition, compliance planning, implementation, and post certification activities.

a. Within the conceptual design phase, the applicant and FAA engage in a "process orientation" meeting where the applicant discusses the objective of the project, and the FAA explains to the applicant the specific requirements to be met. The FAA will provide the applicant with "pre-project guidance" on how to meet FAA requirements. The applicant with provide the FAA with "familiarization briefings" to familiarize the FAA with the proposed design as it is currently known. And the applicant is to provide the FAA with a "certification

[19] FAA Order 8110.4C, Figure 2-1.

plan" with the extent and depth of the information sufficient to determine the feasibility of the applicant's proposed schedule.[20]

b. Within the requirements definition phase, the applicant will submit an application for FAA design approval and develop and submit a "Project Specific Certification Plan" (PSCP).[21] In this phase there is substantial interaction between the applicant and the FAA, as this is the "contract for certification" that is made between the parties. It is within the PSCP the planning for certification of airborne software will be identified to include compliance with DO-178C, as relevant.

c. Within the compliance planning phase, the applicant and FAA establish an agreement as to what level the FAA intends to be involved in the certification project.[22]

d. The implementation phase incorporates the actual witnessing of tests and findings of compliance.[23] Within this phase test plans are carried out and results are documented in reports. When compliance to the specific regulatory requirements has been demonstrated, FAA design approval may be granted.

e. The post-certification activities phase includes those activities that monitor the design (airborne software) as it performs in operations and provides for process that ensure continued airworthiness.[24]

30. In the description of the above, the FAA requires the applicant to submit documents and information in a structured and defined manner, frequently on FAA forms and in formats they are accustomed to receiving. The structure and content of the planning and reports for the certification of airborne software are mostly prescribed by FAA policy or DO-178C.

[20] FAA Order 8110.4C, Section 2-3.

[21] FAA Order 8110.4C, Section 2-4.

[22] FAA Order 8110.4C, Section 2-5.

[23] FAA Order 8110.4C, Section 2-6.

[24] FAA Order 8110.4C, Section 2-7.

B. Industry Standards for Software Development and Documentation

31. In 1935, the Radio Technical Commission for Aeronautics (RTCA) was established to enhance the standardization of electronics within the aviation industry. Since its founding, and in cooperation with global civil aviation authorities, RTCA has evolved into the primary consensus standards developer for avionics and airborne software and is the publisher of the "DO" series of standards.

32. The Institute of Electrical and Electronics Engineers (IEEE) was founded in 1963, with a focus on electronics technical professionals. In addition to providing for professional development, IEEE also publishes consensus standards that relate to product testing and certification, sometimes overlapping to the standards of RTCA. Where RTCA focuses predominately on the aerospace industry, IEEE applies to all industries.

33. As an international consensus standard, DO-178C provides requirements for the certification of software embedded airborne systems and equipment installed on aircraft, engines, propellers. DO-178C was released in January 2012, updating, and replacing DO-178B that was initially published in December 1992. As previously discussed in this declaration, DO-178C is the default acceptable means for compliance for airborne software certification, prescribed and accepted by the FAA and most every other civil aviation authority in the world.

34. International consensus standards such as DO-178C are vitally important for aviation safety as they provide standardized methods for planning, testing, and finding of compliance. Such standardization removes variance from the global supply chain and increases understanding of processes and methods.

35. Consistent with the previously discussed FAA's life-cycle model, DO-178C also applies a standardized life-cycle model for the certification of airborne software, including system safety assessment and validation processes.

DO-178C also provides common terms and definitions that have shared meaning and application throughout the aviation industry. Figure 2 illustrates the DO-178C life-cycle for the certification of airborne software and the interrelationships for planning and reports.

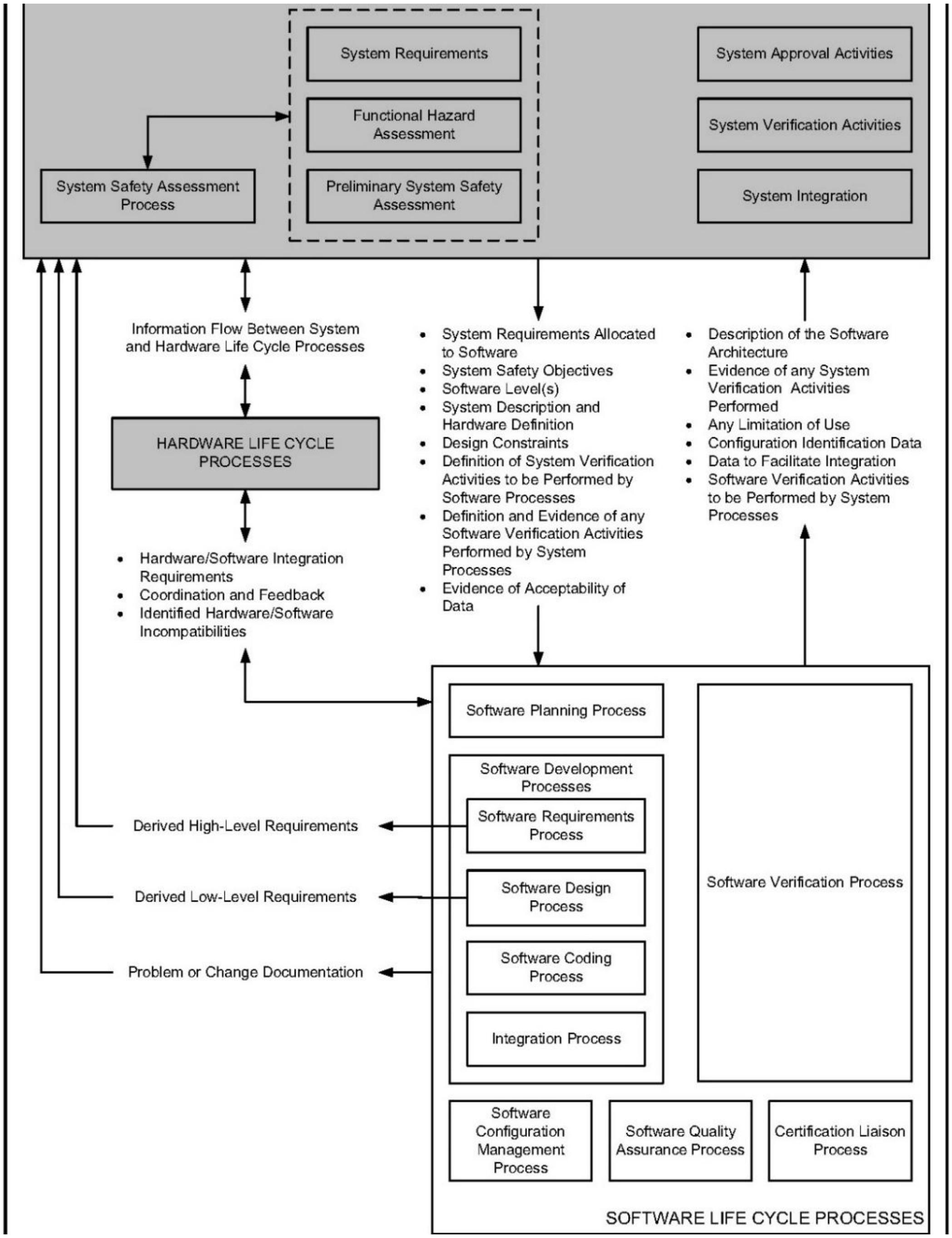

System Requirements
Functional Hazard Assessment
Preliminary System Safety Assessment
System Safety Assessment Process
System Approval Activities
System Verification Activities
System Integration
Information Flow Between System and Hardware Life Cycle Processes
HARDWARE LIFE CYCLE PROCESSES
• Hardware/Software Integration Requirements
• Coordination and Feedback
• Identified Hardware/Software Incompatibilities
• System Requirements Allocated to Software
• System Safety Objectives
• Software Level(s)
• System Description and Hardware Definition
• Design Constraints
• Definition of System Verification Activities to be Performed by Software Processes
• Definition and Evidence of any Software Verification Activities Performed by System Processes
• Evidence of Acceptability of Data
• Description of the Software Architecture
• Evidence of any System Verification Activities Performed
• Any Limitation of Use
• Configuration Identification Data
• Data to Facilitate Integration
• Software Verification Activities to be Performed by System Processes
Derived High-Level Requirements
Derived Low-Level Requirements
Problem or Change Documentation
Software Planning Process
Software Development Processes
Software Requirements Process
Software Design Process
Software Coding Process
Integration Process
Software Verification Process
Software Configuration Management Process
Software Quality Assurance Process
Certification Liaison Process
SOFTWARE LIFE CYCLE PROCESSES

Figure 2., Information flow between system and software life-cycle process.[25]

[25] DO-178C, Figure 2-1.

36. DO-178C prescribes five software plans that are required to satisfy the objectives of the certification standard, and which form the basis of a substantial portion of Mr. Pixley and Mr. Crozier's opinions regarding the use of Moog's purportedly non-public information. As provided in paragraph 4.3. of the standard, "[t]he software plans are:

- The **Plan for Software Aspects of Certification** (*see* 11.1) serves as the primary means for communicating the proposed development methods to the certification authority for agreement, and defines the means of compliance with this document.
- The **Software Development Plan** (*see* 11.2) defines the software life cycle(s), software development environment, and the means by which the software development process objectives will be satisfied.
- The **Software Verification Plan** (*see* 11.3) defines the means by which the software verification process objectives will be satisfied.
- The **Software Configuration Management Plan** (*see* 11.4) defines the means by which the software configuration management process objectives will be satisfied.
- The **Software Quality Assurance Plan** (*see* 11.5) defines the means by which the software quality assurance process objectives will be satisfied."

37. In addition to the above, DO-178C provides for the utilization of checklists to ensure reviews and analysis provide for repeatable evidence and correctness and reviews provide a qualitative assessment of correctness.[26]

Plan for Software Aspects of Certification (PSAC)

38. The "Plan for Software Aspects of Certification" (PSAC) is the foundational planning document associated with avionics software certification. The PSAC is intended to be submitted to, and approved by, certification authorities or a customer acting as the certification authority prior to initiating formal development of avionics software. Related checklist should affirm the PSAC contains sufficient information for the reader to understand system basics along

[26] DO-178C, paragraph 6.3.

with an overview of the system, hardware, software, Design Assurance Level (DAL), and key software engineering lifecycle activities pursuant to DO-178C compliance.

39. The PSAC serves as the foundational plan in describing how safety and DO-178C compliance or certification is to be achieved for a specific airborne software installed in an avionics system. It should provide the reader with a degree of confidence that the applicant understands DO-178C necessities and has made sufficient and correct decisions regarding the ensuing avionics software engineering.

40. Typically, the safety assessment and system requirement definition is complete prior to submitting the PSAC to certification authorities for review and approval; this is because the PSAC must summarize justification for the Design Assurance Level (DAL), *e.g.* "criticality level" of the associated system/software and the system architecture. The applicant should have completed initial software tool selection and nominally completed the following documents, so they can be referenced within the PSAC.

41. The PSAC should also summarize key software components, plus any planned deviations from strict DO-178C interpretation including previously developed software, mixed criticality, reverse-engineering, or alternate means of compliance. Also, the PSAC should summarize any use of Model Based Development, Object Oriented Technology and Formal Methods.

42. As provided in DO-178C, the structure of the PSAC should include:

- System overview;
- Software overview;
- Certification considerations;
- Software life-cycle;
- Software life-cycle data;
- Schedule;

- Additional considerations; and
- Supplier oversight.[27]

Software Development Plan (SDP)

43. The Software Development Plan (SDP) "is a description of the software development procedures and software life cycle(s) to be used to satisfy the software development process objectives. It may be included in the Plan for Software Aspects of Certification." The structure of the SDP is provided in DO-178C, and the plan should include:

> Standard: Identification of the Software Requirements Standards, Software Design Standards, and Software Code Standards for the project.
>
> Software life-cycle: A description of the software life cycle processes to be used to form the specific software life cycle(s) to be used on the project, including the transition criteria for the software development processes.
>
> Software development environment: A statement of the chosen software development environment in terms of hardware and software, "including:
>
> 1. The requirements development method(s) and tools to be used.
> 2. The design method(s) and tools to be used.
> 3. The coding method(s), programming language(s), coding tool(s) to be used, and when applicable, options and constraints of autocode generators.
> 4. The compilers, linkage editors, and loaders to be used.
> 5. The hardware platforms for the tools to be used."[28]

[27] DO-178C, paragraph 11.1.

[28] DO-178C, paragraph 11.2.

Software Verification Plan (SVP)

44. As provided in DO-178C, the Software Verification Plan (SVP) is a description of the verification procedures to be used to satisfy the software verification process objectives. The SVP should include:

a. Organization: Organizational responsibilities within the software verification process and interfaces with the other software life cycle processes.

b. Independence: A description of the methods for establishing verification independence, when required.

c. Verification methods: A description of the verification methods to be used for each activity of the software verification process.

d. Verification environment: A description of the equipment for testing, the testing and analysis tools, and how to apply these tools and hardware test equipment.

e. Transition criteria: The transition criteria for entering the software verification process.

f. Partitioning considerations: If partitioning is used, the methods used to verify the integrity of the partitioning.

g. Compiler assumptions: A description of the assumptions made by the applicant about the correctness of the compiler, linkage editor, or loader.

h. Reverification method: For software modification, a description of the methods for identifying, analyzing, and verifying the affected areas of the software and the changed parts of the Executable Object Code.

i. Previously developed software: For previously developed software, if the initial compliance baseline for the verification process does not comply with this document, a description of the methods to satisfy the objectives of this document.

j. Multiple-version dissimilar software: If multiple-version dissimilar software is used, a description of the software verification process activities.[29]

[29] DO-178C, paragraph 11.3.

Software Configuration Management Plan (SCMP)

45. As provided in DO-178C, "[t]he Software Configuration Management Plan establishes the methods to be used to achieve the objectives of the SCM process throughout the software life cycle." The SCM should include:

a. Environment: A description of the SCM environment to be used, including procedures, tools, methods, standards, organizational responsibilities, and interfaces.

b. Activities: A description of the SCM process activities in the software life cycle:

i. Configuration identification: Items to be identified, when they will be identified, the identification methods for software life cycle data (for example, part numbering), and the relationship of software identification and the system or equipment identification.

ii. Baselines and traceability: The means of establishing baselines, what baselines will be established, when these baselines will be established, the software library controls, and the configuration item and baseline traceability.

iii. Problem reporting: The content and identification of Problem Reports for the software product and software life cycle processes, when they will be written, the method of closing Problem Reports, and the relationship to the change control activity.

iv. Change control: Configuration items and baselines to be controlled, when they will be controlled, the problem/change control activities that control them, precertification controls, post-certification controls, and the means of preserving the integrity of baselines and configuration items.

v. Change review: The method of handling feedback from and to the software life cycle processes; the methods of assessing and prioritizing problems, approving changes, and handling their resolution or change implementation; and the relationship of these methods to the problem reporting and change control activities.

vi. Configuration status accounting: The data to be recorded to enable reporting configuration management status, definition of where that data will be kept, how it will be retrieved for reporting, and when it will be available.

vii. Archive, retrieval, and release: The integrity controls, the release method and authority, and data retention.

viii. Software load control: A description of the software load control safeguards and records.

ix. Software life-cycle environment controls: Controls for the tools used to develop, build, verify, and load the software, addressing sections 11.4.b.1 through 11.4.b.7 of the Standard. This includes control of tools to be qualified.

x. Software life-cycle data controls: Controls associated with CC1 and CC2 data.

c. Transition criteria: The transition criteria for entering the SCM process.

d. SCM data: A definition of the software life cycle data produced by the SCM process, including SCM Records, the Software Configuration Index, and the Software Life Cycle Environment Configuration Index.

e. Supplier control: The means of applying SCM process requirements to suppliers.[30]

Software Quality Assurance Plan (SQAP)

46. As provided in DO-178C, "[t]he Software Quality Assurance Plan establishes the methods to be used to achieve the objectives of the SQA process. The SQA Plan may include descriptions of process improvement, metrics, and progressive management methods." The SQAP should include:

a. Environment: A description of the SQA environment, including scope, organizational responsibilities and interfaces, standards, procedures, tools, and methods.

b. Authority: A statement of the SQA authority, responsibility, and independence, including the SQA approval of software products.

[30] DO-178C, paragraph 11.4.

c. Activities: The SQA activities that are to be performed for each software life cycle process and throughout the software life cycle including:

1. SQA methods, for example, reviews, audits, reporting, inspections, and monitoring of the software life cycle processes.
2. Activities related to the problem reporting, tracking, and corrective action system.
3. A description of the software conformity review activity.

d. Transition criteria: The transition criteria for entering the SQA process.

e. Timing: The timing of the SQA process activities in relation to the activities of the software life cycle processes.

f. SQA Records: A definition of the records to be produced by the SQA process.

g. Supplier oversight: A description of the means of ensuring that suppliers' processes and outputs will comply with the plans and standards.[31]

47. As with any consensus standard, DO-178C is authored by industry and information and guidance for compliance is readily and publicly available. The objective of industry consensus standards such as DO-178C is to reduce variation throughout industry and increase the level of confidence in safety that complied with relevant airworthiness requirements.[32]

48. As described above, the structure and content of airborne software certification documentation is widely standardized as such must follow the requirements of DO-178C. Any variation from the standardized structure would cause a pause by the FAA and require additional focused review to find equivalence of safety.

[31] DO-178C, paragraph 11.6.

[32] DO-178C, paragraph 11.1.

C. Real-Time Operating Systems (RTOS)

49. As part of the testing of airborne software, DO-178C requires planning and testing to ensure software and hardware integration meet requirements for "definition of protocols, timing constraints, and addressing schemes."[33] As a tool, aviation software design organizations routinely employ Real-Time Operating Systems (RTOS) to test the software and hardware interfaces.

50. An RTOS is an operating system (OS) for real-time computing applications that processes data and events that have critically defined time constraints. An RTOS is distinct from a time-sharing operating system, such as Unix, which manages the sharing of system resources with a scheduler, data buffers, or fixed task prioritization in a multitasking or multiprogramming environment. Processing time requirements need to be fully understood and bound rather than just kept as a minimum. All processing must occur within the defined constraints. Real-time operating systems are event-driven and preemptive, meaning the OS can monitor the relevant priority of competing tasks, and make changes to the task priority. Event-driven systems switch between tasks based on their priorities, while time-sharing systems switch the task based on clock interrupts.

51. The objective of utilizing an RTOS in the validation and certification processes of airborne software is to simulate the actual use of the software in a model that represents the aircraft and/or system. There is a very broad selection of RTOS OS available, much of it is open-source and available for free download.[34] However, not all RTOS OS are equally appropriate for the application with

[33] DO-178C, paragraph 2.2.3. *See also* paragraphs 6.3.4, 11.1.b, and 11.9.d and f.

[34] https://en.wikipedia.org/wiki/Comparison_of_real-time_operating_systems.

airborne software. The FAA has provided both research and guidance on what applicants should consider when selecting an RTOS OS.[35]

52. As provided in FAA AC 20-115D, RTCA standards DO-330 and DO-331 are acceptable means of showing regulatory compliance for qualification of software tools, including RTOS OS. Applicants are required to incorporate the tool qualification planning for the RTOS as part of the PSAC.

IV. OPINIONS

A. Methodology

53. Both Mr. Pixley and Mr. Crozier claim to have identified examples of Moog non-public information that was used by Skyryse after March 11, 2022 or which Skyryse failed to produce after April 1, 2022.[36] However, neither Mr. Crozier nor Mr. Pixley provide a concrete definition of what actually constitutes "Moog non-public information," in their declarations or at deposition.

54. I am not a lawyer and I do not intend to give any sort of legal meaning or interpretation to the term "Moog non-public information." I am experienced in aviation, and have become familiar over the years with how companies in the industry treat their sensitive, proprietary information. For purposes of this declaration, when considering whether a document might be "Moog non-public information" as Mr. Crozier and Mr. Pixley contend, I considered whether there were any factual indications that information is, in fact, Moog's and not originated or derived from a non-Moog source, and also whether it is publicly available or derived from a publicly available source or generally known to the public.

55. I understand that both Mr. Pixley and Mr. Crozier confirmed [redacted] I

[35] Reference Report DOT/FAA/AR-03/77, Commercial off-the-shelf Real-Time Operating System and Architectural Considerations, dated February 2004. (https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/media/03-77_COTS_RTOS.pdf)

[36] *See, e.g.*, Pixley Decl. ¶¶ 22; Crozier Decl. ¶¶ 11-15.

understand that Mr. Pixley stated that he [redacted][37] I also understand Mr. Crozier similarly confirmed that he [redacted][40]

56. Instead, I understand that to support their assertions that certain information is Moog non-public information, Moog's experts rely on the fact [redacted].[41]

57. Based on my experience working with companies in the aviation industry, Mr. Pixley and Mr. Crozier's methodology provides an insufficient basis to conclude whether a document or its contents actually constitute Moog nonpublic information. For example, Mr. Pixley confirmed that he [redacted]

[37] *See* Pixley [redacted]

[38] Crozier Dep. 79:4-15.

[39] Crozier Dep. 20:2-9.

[40] Crozier Dep. 32:9-11

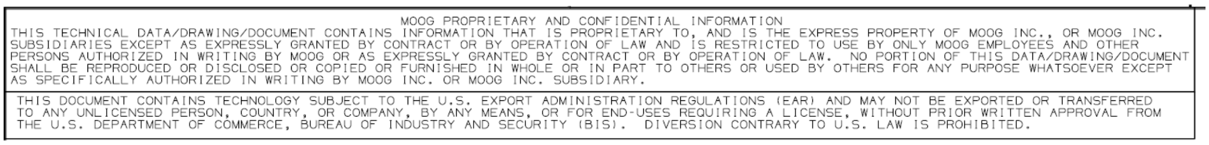
[41] Crozier Dep. 22:13-23:23, 87:14-25; Pixley Dep. 20:2-20.

[42] Pixley Dep. 24:10-21.

[43] Pixley Dep. 22:8-14.

[redacted][45]

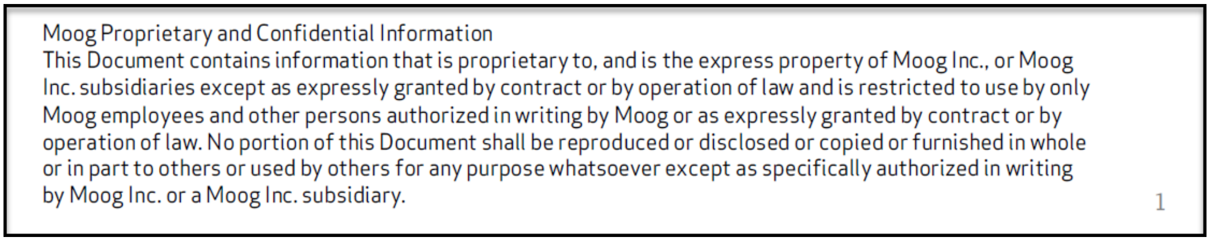
58. Included below are images of similar Moog confidentiality and proprietary legends contained in documents that are publicly available, including on the internet and on Moog's own website, and thus do not appear to be treated as confidential or proprietary or non-public in practice. Figures 3-5 are publicly available through an internet search while Figure 6 is identified by Mr. Crozier on a document he describes as Moog nonpublic information.



MOOG PROPRIETARY AND CONFIDENTIAL INFORMATION
THIS TECHNICAL DATA/DRAWING/DOCUMENT CONTAINS INFORMATION THAT IS PROPRIETARY TO, AND IS THE EXPRESS PROPERTY OF MOOG INC., OR MOOG INC. SUBSIDIARIES EXCEPT AS EXPRESSLY GRANTED BY CONTRACT OR BY OPERATION OF LAW AND IS RESTRICTED TO USE BY ONLY MOOG EMPLOYEES AND OTHER PERSONS AUTHORIZED IN WRITING BY MOOG OR AS EXPRESSLY GRANTED BY CONTRACT OR BY OPERATION OF LAW. NO PORTION OF THIS DATA/DRAWING/DOCUMENT SHALL BE REPRODUCED OR DISCLOSED OR COPIED OR FURNISHED IN WHOLE OR IN PART TO OTHERS OR USED BY OTHERS FOR ANY PURPOSE WHATSOEVER EXCEPT AS SPECIFICALLY AUTHORIZED IN WRITING BY MOOG INC. OR MOOG INC. SUBSIDIARY.

THIS DOCUMENT CONTAINS TECHNOLOGY SUBJECT TO THE U.S. EXPORT ADMINISTRATION REGULATIONS (EAR) AND MAY NOT BE EXPORTED OR TRANSFERRED TO ANY UNLICENSED PERSON, COUNTRY, OR COMPANY, BY ANY MEANS, OR FOR END-USES REQUIRING A LICENSE, WITHOUT PRIOR WRITTEN APPROVAL FROM THE U.S. DEPARTMENT OF COMMERCE, BUREAU OF INDUSTRY AND SECURITY (BIS). DIVERSION CONTRARY TO U.S. LAW IS PROHIBITED.

Figure 3., 72_Installation_Drawing_CA79668_F[46]



Moog Proprietary and Confidential Information
This Document contains information that is proprietary to, and is the express property of Moog Inc., or Moog Inc. subsidiaries except as expressly granted by contract or by operation of law and is restricted to use by only Moog employees and other persons authorized in writing by Moog or as expressly granted by contract or by operation of law. No portion of this Document shall be reproduced or disclosed or copied or furnished in whole or in part to others or used by others for any purpose whatsoever except as specifically authorized in writing by Moog Inc. or a Moog Inc. subsidiary.

1

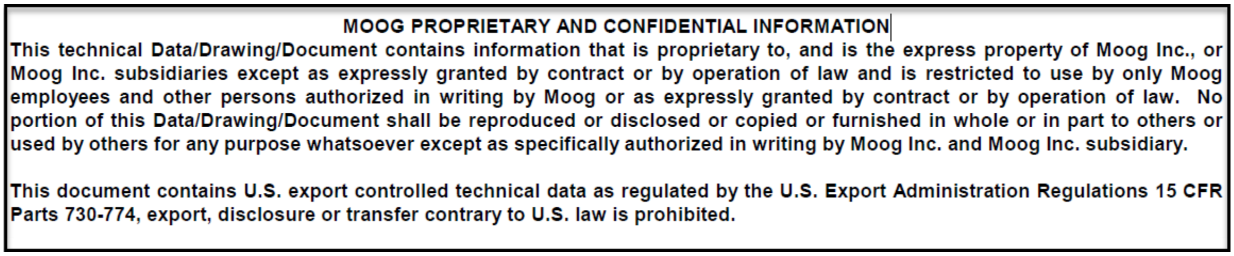
Figure 4., Moog Test Software Supported Hardware v3.24[47]

[44] Pixley Dep. 23:12-19.

[45] Pixley Dep. 24:1-9.

[46] Ex. D5, Found publicly at https://www.moog.com/content/dam/moog/literature/ICD/72_Installation_Drawing_CA79668_F.pdf Last accessed on April 23, 2023.

[47] Ex. D6, Found publicly at https://www.moog.com/content/dam/sites/moog/images/Markets/simulation-&-test/test/Moog%20Test%20Software%20Supported%20Hardware%20v3.24.pdf. Last accessed on April 23, 2023.



MOOG PROPRIETARY AND CONFIDENTIAL INFORMATION

This technical Data/Drawing/Document contains information that is proprietary to, and is the express property of Moog Inc., or Moog Inc. subsidiaries except as expressly granted by contract or by operation of law and is restricted to use by only Moog employees and other persons authorized in writing by Moog or as expressly granted by contract or by operation of law. No portion of this Data/Drawing/Document shall be reproduced or disclosed or copied or furnished in whole or in part to others or used by others for any purpose whatsoever except as specifically authorized in writing by Moog Inc. and Moog Inc. subsidiary.

This document contains U.S. export controlled technical data as regulated by the U.S. Export Administration Regulations 15 CFR Parts 730-774, export, disclosure or transfer contrary to U.S. law is prohibited.

Figure 5., MRM53618 REV[48]

[redacted]

Figure 6 – Crozier Ex. C-4

59. As described below, this is a flaw that permeates Mr. Pixley and Mr. Crozier's opinions and undermines their conclusions that Skyryse used Moog non-public information after March 11, 2022 or failed to produce it after April 1, 2022.

60. As one example, Mr. Pixley writes in Paragraphs 8 and 9 of his declaration that he conducted a search for files on a laptop issued to Skyryse employee, Sathya Achar. He asserts that he "reviewed a folder in the root directory" in which he "conducted a simple search of files" for the word "Moog."[49] According to Mr. Pixley, that "search resulted in 81 Office-type documents (Word, Excel, PowerPoint) that contained 'Moog Inc.' or Moog" in the company metadata field; one PDF document that contained the line "MOOG PROPRIETARY AND CONFIDENTIAL INFORMATION" and, 173 PDF documents that contained on of the following lines of text: 'Material licensed to Moog Inc;' 'Sold to MOOG INC;' 'Downloaded by Moog Inc;' and, 'Issued to Moog Inc.;'" and Mr. Pixley claims that "[s]ome of these types of documents included a non-public, proprietary

[48] Ex. D7, Found publicly at https://www.moog.com/content/dam/moog/literature/Corporate/Suppliers/p-b/MRM53618%20REV.pdf. Last accessed on April 23, 2023.

[49] Pixley Decl. ¶ 9.

flight control document for the Airbus A350, required performance information, and intermediate work-in-progress."[50]

61. I have reviewed the documents described in Paragraph 9 of Mr. Pixley's declaration and nearly all of them are standards related documents that are publicly available, including for purchase. The one flight control document for the Airbus A350 referenced by Mr. Pixley is a Plan for Hardware Aspects of Certification, a type of document I describe in further detail below and the template for which is based on the relevant DO-178C standards.

62. As another example, in Paragraphs 46-49 of his declaration, Mr. Pixley describes a file he claims was sent by former Skyryse employee Reid Raithel to his colleagues at Skyryse, which he asserts contains the word "Moog Inc." in the Company metadata field and which he claims "appears to be a recruiting list of targeted Moog employees."[51] Mr. Pixley does not directly claim that this document constitutes Moog non-public information but he seems to imply it. Nonetheless, I have reviewed the document, which contains publicly available information available on LinkedIn. Moreover, a close review of the individuals on the list confirms that not all of them were Moog employees, which fact can be deduced by clicking on the hyperlinks for the LinkedIn pages contained in the spreadsheet.

B. DPA/Software Process Checklists.

63. Mr. Crozier opines that Skyryse and its personnel used and exchanged certain checklists, including checklists he refers to as DPA Review Checklists, SW Document Review Checklists, and software process checklists, which he refers to as "Moog Non-Public Information."[52] Mr. Pixley also refers to similar checklists

[50] Pixley Decl. ¶ 9

[51] Pixley Decl. ¶¶ 47-48.

[52] Crozier Decl. at 7, ¶¶ 19-23, 31-32, 40-42, 44.

as "examples of Moog non-public data that was transmitted by Skyryse email accounts."[53]

64. The software checklists Mr. Crozier and Mr. Pixley identify are exemplars of the types of checklists that are commonly used in industry and are widely available. Such software checklists that are structured around compliance with the RTCA standards, particularly DO-178, and DO-330 and other related standards, that are widely used by companies in the aviation industry to prepare for certification audits and are readily available for purchase from third parties. Such third-party offerings contain headers and checklist items that, in many cases, reflect virtually identical substantive content and even text that matches verbatim the checklists Moog's experts considered.

65. This makes sense because all these checklists are derived from the same set of standards that are globally used by companies in the industry. It would make no sense for each company to create their own unique checklists, which would severely complicate the auditing and certification processes. Instead, companies are expected to follow similar formatting, structure, and design consistent with the relevant standards so that information is delivered in a straightforward and easy to understand fashion.

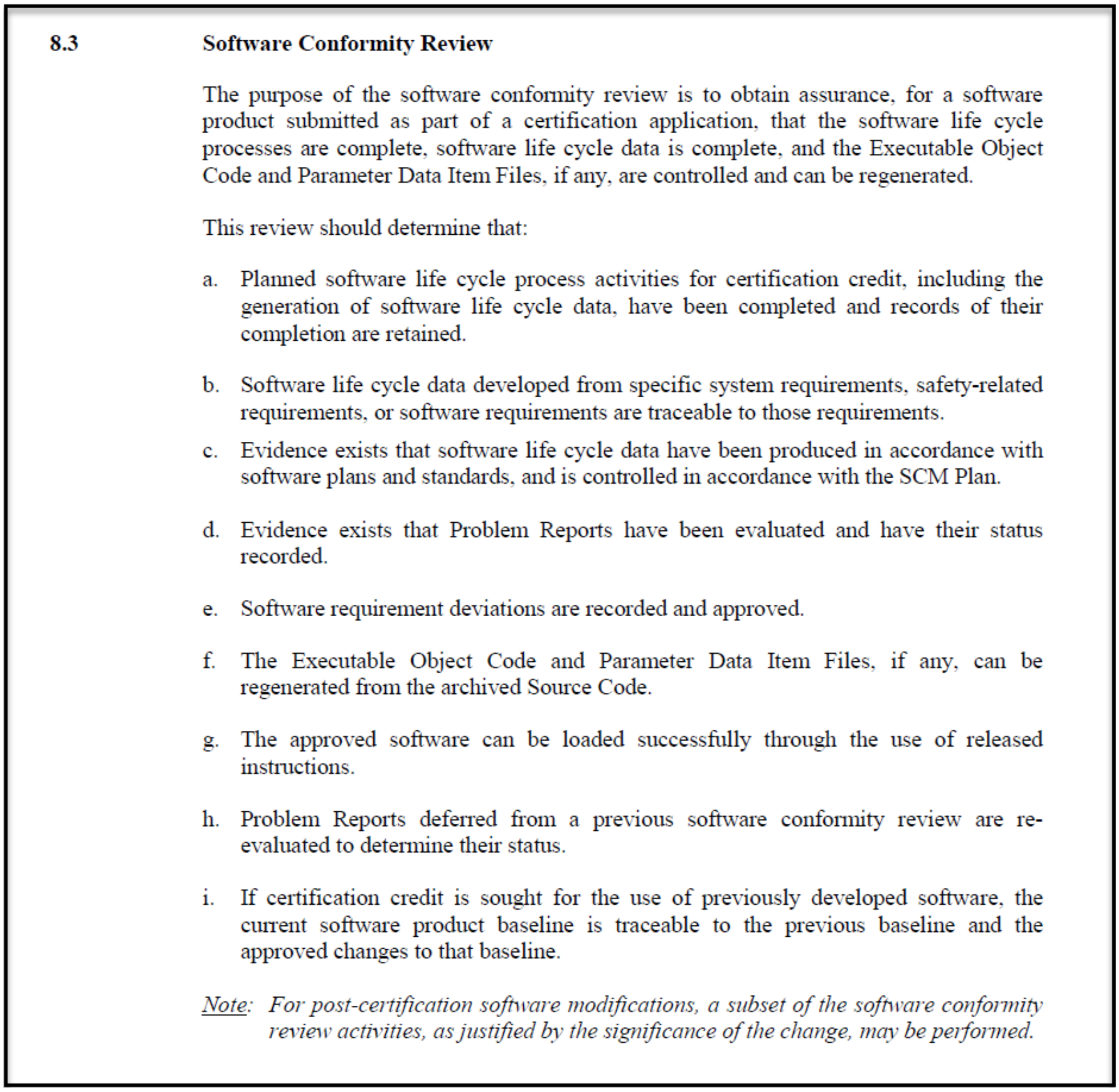
66. A comparison of the contents of the checklists identified by Moog's experts to publicly available checklists and relevant standards documents shows that the checklists identified by Mr. Crozier and Mr. Pixley contain publicly available information and thus do not appear to me to be unique to Moog. While the below only reflects a handful of examples, the parallels between publicly available information and what Mr. Pixley and Mr. Crozier describe as Moog's non-public information that was "misappropriated" by Skyryse is unmistakable. Figure 7 reflects a screenshot from a Moog checklist identified by Mr. Crozier as

[53] *See, e.g.*, Pixley Decl. ¶¶ 22, 25-27, 30, 34-36, 40.

Moog nonpublic information. Figure 8 reflects the DO-178C standards on which Figure 7 is based, and Figure 9 reflects a similar checklist available from a third-party source.

[redacted]

Figure 7., Crozier Ex. A-2, [redacted]



8.3 Software Conformity Review

The purpose of the software conformity review is to obtain assurance, for a software product submitted as part of a certification application, that the software life cycle processes are complete, software life cycle data is complete, and the Executable Object Code and Parameter Data Item Files, if any, are controlled and can be regenerated.

This review should determine that:

a. Planned software life cycle process activities for certification credit, including the generation of software life cycle data, have been completed and records of their completion are retained.

b. Software life cycle data developed from specific system requirements, safety-related requirements, or software requirements are traceable to those requirements.

c. Evidence exists that software life cycle data have been produced in accordance with software plans and standards, and is controlled in accordance with the SCM Plan.

d. Evidence exists that Problem Reports have been evaluated and have their status recorded.

e. Software requirement deviations are recorded and approved.

f. The Executable Object Code and Parameter Data Item Files, if any, can be regenerated from the archived Source Code.

g. The approved software can be loaded successfully through the use of released instructions.

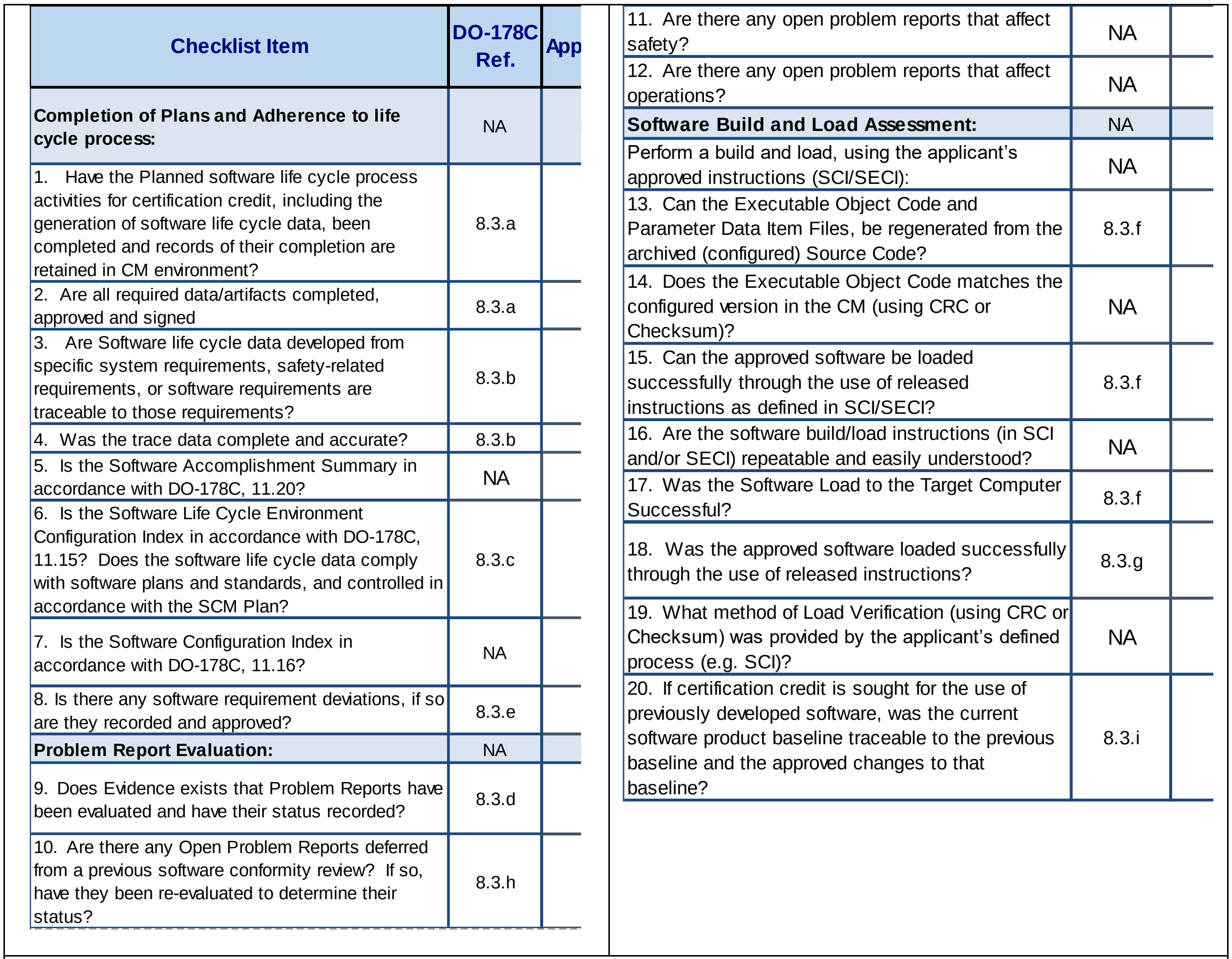
h. Problem Reports deferred from a previous software conformity review are re-evaluated to determine their status.

i. If certification credit is sought for the use of previously developed software, the current software product baseline is traceable to the previous baseline and the approved changes to that baseline.

<u>Note</u>: *For post-certification software modifications, a subset of the software conformity review activities, as justified by the significance of the change, may be performed.*

Figure 8., Publicly Available 2011 DO-178C – Section 8.3 –Software Conformity Review[54]

[54] Ex. D2.



Checklist Item	DO-178C Ref.	App
Completion of Plans and Adherence to life cycle process:	NA	
1. Have the Planned software life cycle process activities for certification credit, including the generation of software life cycle data, been completed and records of their completion are retained in CM environment?	8.3.a	
2. Are all required data/artifacts completed, approved and signed	8.3.a	
3. Are Software life cycle data developed from specific system requirements, safety-related requirements, or software requirements are traceable to those requirements?	8.3.b	
4. Was the trace data complete and accurate?	8.3.b	
5. Is the Software Accomplishment Summary in accordance with DO-178C, 11.20?	NA	
6. Is the Software Life Cycle Environment Configuration Index in accordance with DO-178C, 11.15? Does the software life cycle data comply with software plans and standards, and controlled in accordance with the SCM Plan?	8.3.c	
7. Is the Software Configuration Index in accordance with DO-178C, 11.16?	NA	
8. Is there any software requirement deviations, if so are they recorded and approved?	8.3.e	
Problem Report Evaluation:	NA	
9. Does Evidence exists that Problem Reports have been evaluated and have their status recorded?	8.3.d	
10. Are there any Open Problem Reports deferred from a previous software conformity review? If so, have they been re-evaluated to determine their status?	8.3.h	
11. Are there any open problem reports that affect safety?	NA	
12. Are there any open problem reports that affect operations?	NA	
Software Build and Load Assessment:	NA	
Perform a build and load, using the applicant's approved instructions (SCI/SECI):	NA	
13. Can the Executable Object Code and Parameter Data Item Files, be regenerated from the archived (configured) Source Code?	8.3.f	
14. Does the Executable Object Code matches the configured version in the CM (using CRC or Checksum)?	NA	
15. Can the approved software be loaded successfully through the use of released instructions as defined in SCI/SECI?	8.3.f	
16. Are the software build/load instructions (in SCI and/or SECI) repeatable and easily understood?	NA	
17. Was the Software Load to the Target Computer Successful?	8.3.f	
18. Was the approved software loaded successfully through the use of released instructions?	8.3.g	
19. What method of Load Verification (using CRC or Checksum) was provided by the applicant's defined process (e.g. SCI)?	NA	
20. If certification credit is sought for the use of previously developed software, was the current software product baseline traceable to the previous baseline and the approved changes to that baseline?	8.3.i	

Figure 9 Third-Party ConsuNova QA_SW_Confirmity_Audit Checklist[55]

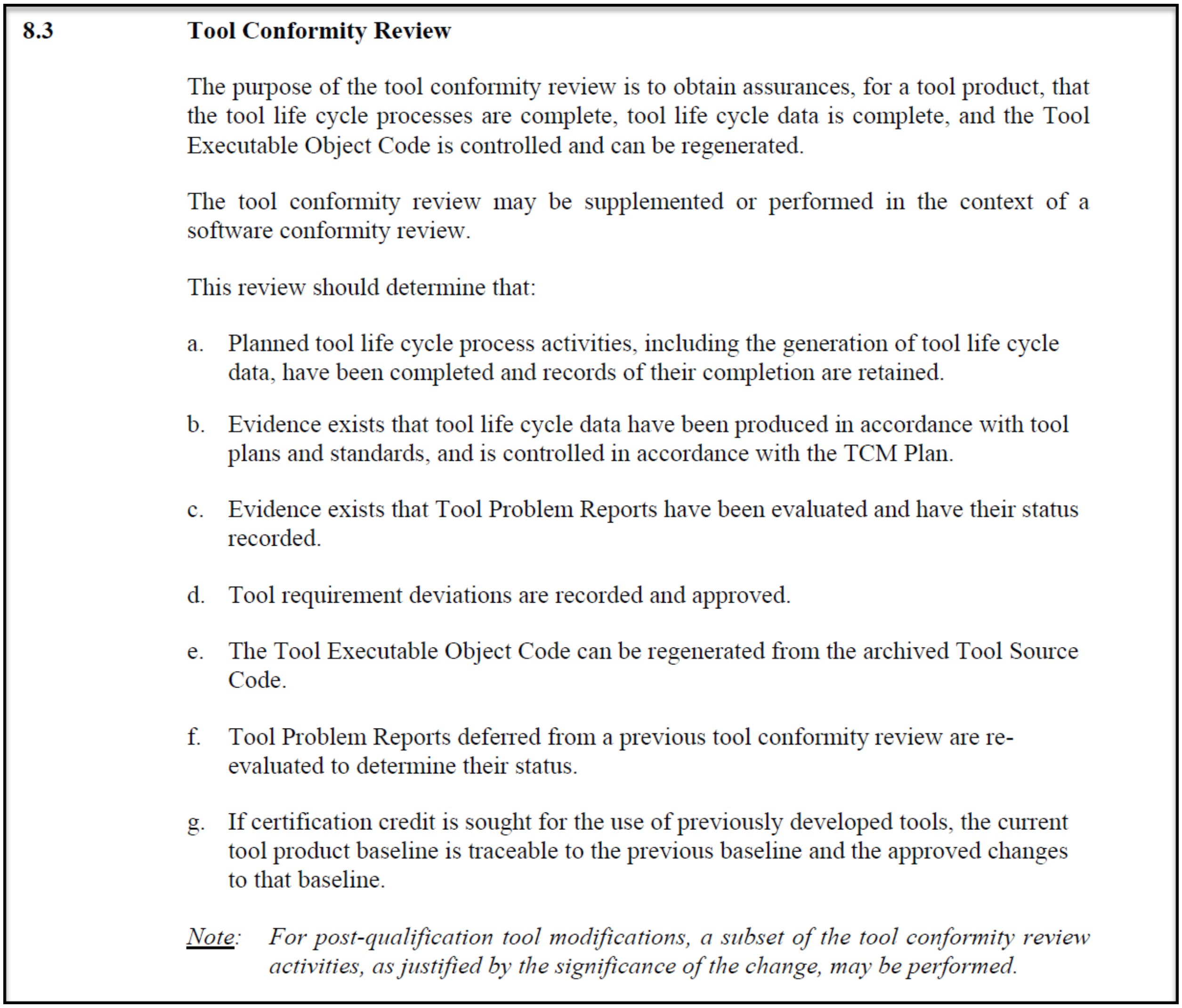
67. Similarly below, Figure 10 reflects a screenshot from a Moog checklist identified by Mr. Crozier as Moog nonpublic information. Figure 11 reflects the DO-178C standards on which Figure 10 is based.

[55] Ex. D15.

[redacted]

Figure 10., Crozier Ex. A-3, [redacted]

[redacted]



8.3 Tool Conformity Review

The purpose of the tool conformity review is to obtain assurances, for a tool product, that the tool life cycle processes are complete, tool life cycle data is complete, and the Tool Executable Object Code is controlled and can be regenerated.

The tool conformity review may be supplemented or performed in the context of a software conformity review.

This review should determine that:

a. Planned tool life cycle process activities, including the generation of tool life cycle data, have been completed and records of their completion are retained.

b. Evidence exists that tool life cycle data have been produced in accordance with tool plans and standards, and is controlled in accordance with the TCM Plan.

c. Evidence exists that Tool Problem Reports have been evaluated and have their status recorded.

d. Tool requirement deviations are recorded and approved.

e. The Tool Executable Object Code can be regenerated from the archived Tool Source Code.

f. Tool Problem Reports deferred from a previous tool conformity review are re-evaluated to determine their status.

g. If certification credit is sought for the use of previously developed tools, the current tool product baseline is traceable to the previous baseline and the approved changes to that baseline.

Note: *For post-qualification tool modifications, a subset of the tool conformity review activities, as justified by the significance of the change, may be performed.*

Figure 11., Publicly Available 2011 DO-330 – Section 8.3 – Tool Conformity Review[56]

68. As reflected above, the checklists Mr. Crozier and Mr. Pixley identify as Moog non-public information are examples of the types of checklists that are commonly used in industry. These checklists, which are structured around compliance with the RTCA standards, particularly the DO-178 and DO-330 standards, and thus have nearly identical content, sometimes verbatim. And even where the precise wording may differ, the substance of the content is necessarily the same or highly similar, which makes sense they are aimed at helping the user or reader comply with the same standards. These types of checklists are widely

[56] Ex. D3, DO-330.

used by companies in the aviation industry to prepare for certification audits and are readily available for purchase from third-parties. Such third-party offerings contain headers and checklist items that, in many cases, reflect virtually identical substantive content and even text that matches verbatim the checklists Mr. Crozier and Mr. Pixley considered to Moog non-public information, which themselves borrowed language verbatim from the DO-standards.

C. Plans for Software Aspects of Certification (PSAC)

69. A PSAC is a document that describes the plan that a company's software engineering team will follow to comply with DO-178C, the guidance that the FAA uses to evaluate and approve airborne civil aviation software. The guidance in DO-178C is standard and many template PSAC's incorporating that guidance are available online for free downloading. There is nothing confidential about that guidance or the various templates that display it. Rather, the bespoke information that each company inputs into the PSAC that details its specific plan for achieving FAA approval for its specific product (such as a commercial airliner versus a helicopter used for general aviation) is what differentiates one PSAC from another.

70. In paragraphs 33 through 35, Mr. Crozier discusses a diagram that he asserts appears in a Moog Plan for Software Aspects of Certification ("PSAC") that he claims reflects the "software part structure" for a Moog project, which he compares to a diagram that appears in a Skyryse PSAC. Mr. Crozier claims the figures are "nearly identical" and constitute "Evidence of Misappropriation in Rex Hyde Production,"[57] suggesting that the format and structure of the PSAC is unique to Moog.[58] However, the illustrated software part structure is widely adopted in industry and examples are publicly available.[59] The definition of the

[57] Crozier Decl. at 13.

[58] Crozier Decl. ¶¶ 33-35 (citing MOOG0030721; BIRD_SR_00011016).

[59] https://ps-engineering.com/Product_Photos/TSO/002-145-1780_PAC45T_PSAC_R1.pdf.

software structure is also a requirement of DO-178C,[60] DO-330,[61] and DO-331.[62] The identifiable similarities between the Moog illustration and that of Skyryse are limited; for instance, they both use boxes and arrows and include the word [redacted] a number of times. As Mr. Crozier admitted, the content in the boxes is distinctly different, including the part numbering scheme.[63]

71. On pages 28 through 30 of Mr. Crozier's declaration, he claims Skyryse copied "word-for-word" Moog's software certification plan table of contents. Mr. Crozier fails to recognize that the outline of the noted table of contents is prescribed by DO-178C[64] and the same format utilized by Moog and Skyryse is widely applied in industry and publicly available.[65] He continues his "word-for-word" opinion on pages 31 and 32 in a comparison of various excerpts from Moog's and Skyryse's PSAC. Again, the entirety of the excerpts are not identical. Skyryse has tailored its compliance to DO-178C and DO-330 differently than Moog. Where the language is identical, the sources are the relevant standards.[66] The way by which both Moog and Skyryse have approached their compliance with requirements is consistent with how others in industry develop their planning documents, which takes the relevant requirements of a standard and incorporates it into the company planning documents.

72. As reflected below, Figure 12 reflects a screenshot of Paragraph 59 of Mr. Crozier's declaration of a table of contents to a Moog PSAC, which table of contents Mr. Crozier opines constitutes Moog's nonpublic information. Figure 13

[60] DO-178C section 6.4.4.2.

[61] DO-330 Appendix D section 1.8.3.

[62] DO-331 sections MB.11.10 and MB.B.17.3-7.

[63] Crozier Dep. 97:16-18.

[64] DO-178C section 11.1.

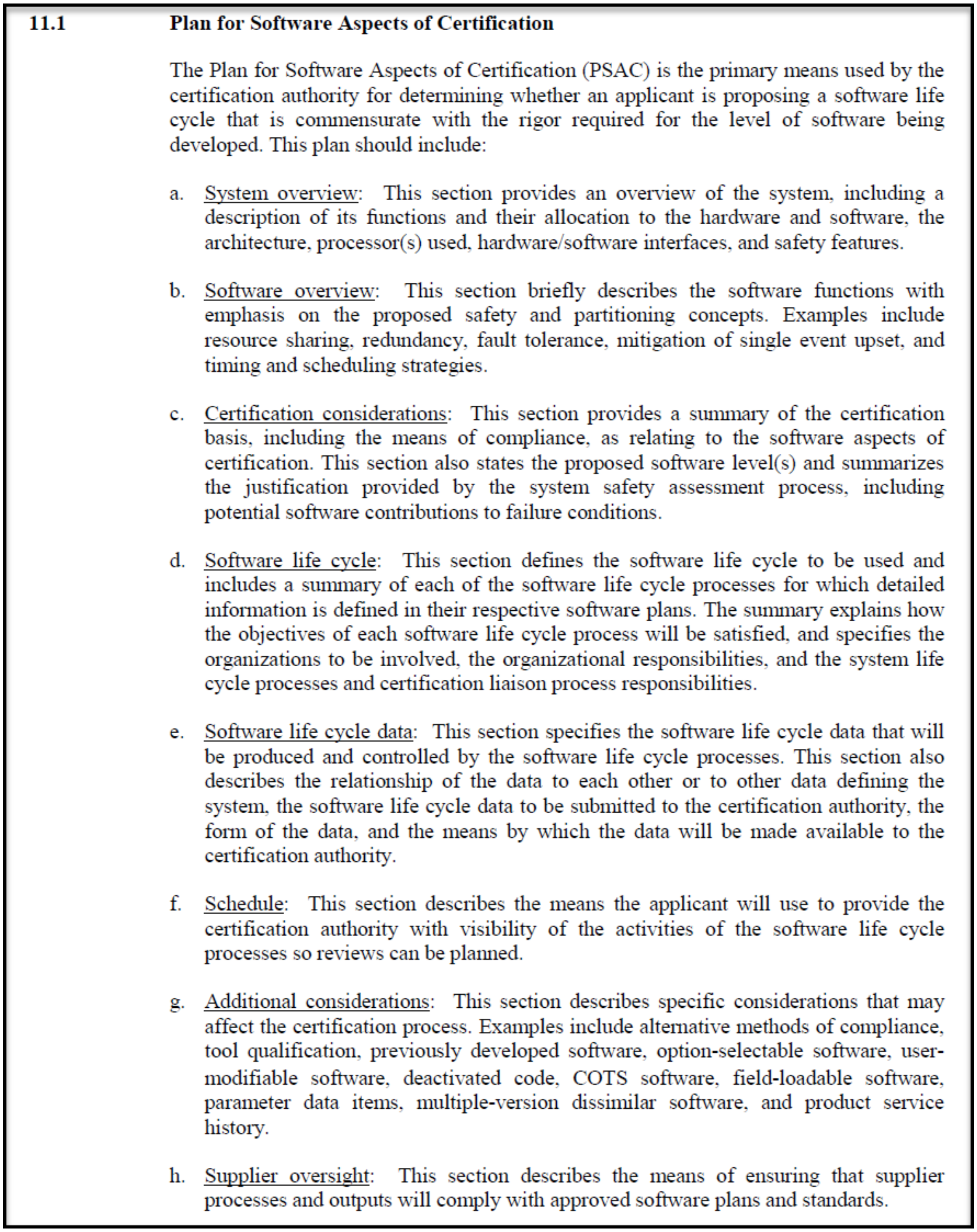
[65] Examples provided: https://ps-engineering.com/Product_Photos/TSO/002-145-1780_PAC45T_PSAC_R1.pdf; https://cgavlas.files.wordpress.com/2018/08/psac-template.pdf; and https://www.drdo.gov.in/centre_for_military_airworthines/templates.

[66] DO-178C section 6.3.4, DO-330 section 10.2.3, and DO-331 section MB.6.3.4.

contains the relevant DO-178C standard on which Figure 12 is based. Figures 14 and 15 reflect screenshots of similar publicly-available tables of contents found through an internet search, and Figure 16 reflects the similar structure of the Skyryse PSAC table of contents.

[redacted]

Figure 12., Crozier Paragraph 59 – [redacted]



11.1 Plan for Software Aspects of Certification

The Plan for Software Aspects of Certification (PSAC) is the primary means used by the certification authority for determining whether an applicant is proposing a software life cycle that is commensurate with the rigor required for the level of software being developed. This plan should include:

a. System overview: This section provides an overview of the system, including a description of its functions and their allocation to the hardware and software, the architecture, processor(s) used, hardware/software interfaces, and safety features.

b. Software overview: This section briefly describes the software functions with emphasis on the proposed safety and partitioning concepts. Examples include resource sharing, redundancy, fault tolerance, mitigation of single event upset, and timing and scheduling strategies.

c. Certification considerations: This section provides a summary of the certification basis, including the means of compliance, as relating to the software aspects of certification. This section also states the proposed software level(s) and summarizes the justification provided by the system safety assessment process, including potential software contributions to failure conditions.

d. Software life cycle: This section defines the software life cycle to be used and includes a summary of each of the software life cycle processes for which detailed information is defined in their respective software plans. The summary explains how the objectives of each software life cycle process will be satisfied, and specifies the organizations to be involved, the organizational responsibilities, and the system life cycle processes and certification liaison process responsibilities.

e. Software life cycle data: This section specifies the software life cycle data that will be produced and controlled by the software life cycle processes. This section also describes the relationship of the data to each other or to other data defining the system, the software life cycle data to be submitted to the certification authority, the form of the data, and the means by which the data will be made available to the certification authority.

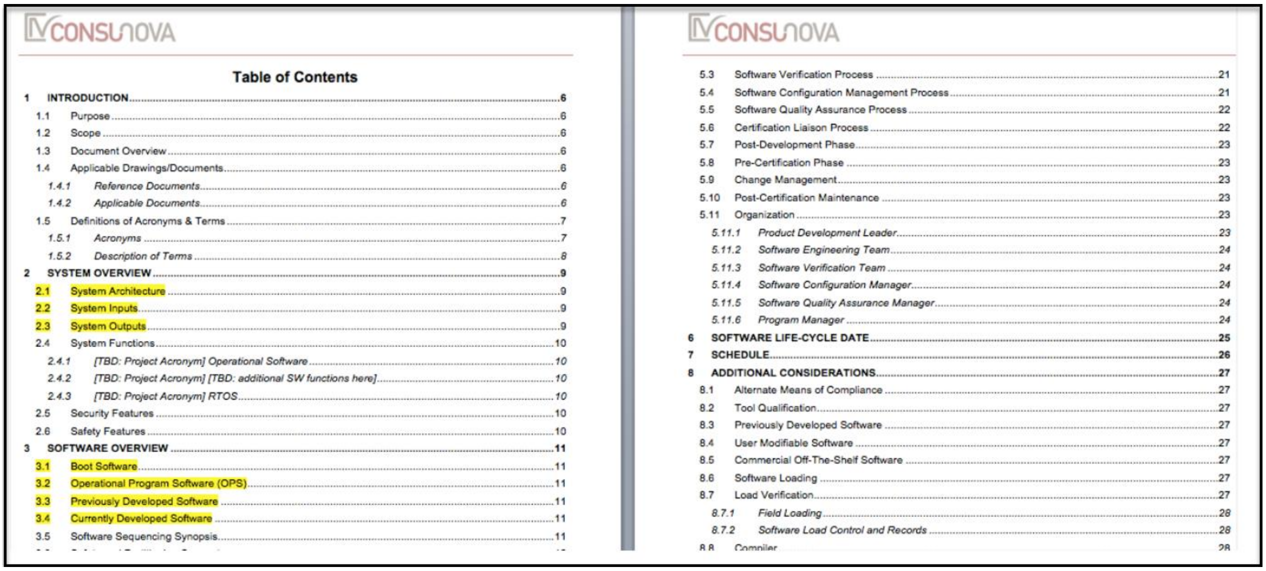
f. Schedule: This section describes the means the applicant will use to provide the certification authority with visibility of the activities of the software life cycle processes so reviews can be planned.

g. Additional considerations: This section describes specific considerations that may affect the certification process. Examples include alternative methods of compliance, tool qualification, previously developed software, option-selectable software, user-modifiable software, deactivated code, COTS software, field-loadable software, parameter data items, multiple-version dissimilar software, and product service history.

h. Supplier oversight: This section describes the means of ensuring that supplier processes and outputs will comply with approved software plans and standards.

Figure 13., Publicly Available DO-178C – Section 11.1 – Plan for Software Aspects of Certification[67]

[67] Ex. D2, DO-178C.



CONSUNOVA

Table of Contents

1 INTRODUCTION 6
1.1 Purpose 6
1.2 Scope 6
1.3 Document Overview 6
1.4 Applicable Drawings/Documents 6
1.4.1 Reference Documents 6
1.4.2 Applicable Documents 6
1.5 Definitions of Acronyms & Terms 7
1.5.1 Acronyms 7
1.5.2 Description of Terms 8
2 SYSTEM OVERVIEW 9
2.1 System Architecture 9
2.2 System Inputs 9
2.3 System Outputs 9
2.4 System Functions 10
2.4.1 [TBD: Project Acronym] Operational Software 10
2.4.2 [TBD: Project Acronym] [TBD: additional SW functions here] 10
2.4.3 [TBD: Project Acronym] RTOS 10
2.5 Security Features 10
2.6 Safety Features 10
3 SOFTWARE OVERVIEW 11
3.1 Boot Software 11
3.2 Operational Program Software (OPS) 11
3.3 Previously Developed Software 11
3.4 Currently Developed Software 11
3.5 Software Sequencing Synopsis 11

CONSUNOVA

5.3 Software Verification Process 21
5.4 Software Configuration Management Process 21
5.5 Software Quality Assurance Process 22
5.6 Certification Liaison Process 22
5.7 Post-Development Phase 23
5.8 Pre-Certification Phase 23
5.9 Change Management 23
5.10 Post-Certification Maintenance 23
5.11 Organization 23
5.11.1 Product Development Leader 23
5.11.2 Software Engineering Team 24
5.11.3 Software Verification Team 24
5.11.4 Software Configuration Manager 24
5.11.5 Software Quality Assurance Manager 24
5.11.6 Program Manager 24
6 SOFTWARE LIFE-CYCLE DATE 25
7 SCHEDULE 26
8 ADDITIONAL CONSIDERATIONS 27
8.1 Alternate Means of Compliance 27
8.2 Tool Qualification 27
8.3 Previously Developed Software 27
8.4 User Modifiable Software 27
8.5 Commercial Off-The-Shelf Software 27
8.6 Software Loading 27
8.7 Load Verification 27
8.7.1 Field Loading 28
8.7.2 Software Load Control and Records 28
8.8 Compiler 28

Figure 14., Publicly Available Third-Party ConsuNova PSAC Template[68]

[68] Found at https://tinyurl.com/4tx5pb67. Last Accessed April 23, 2023.

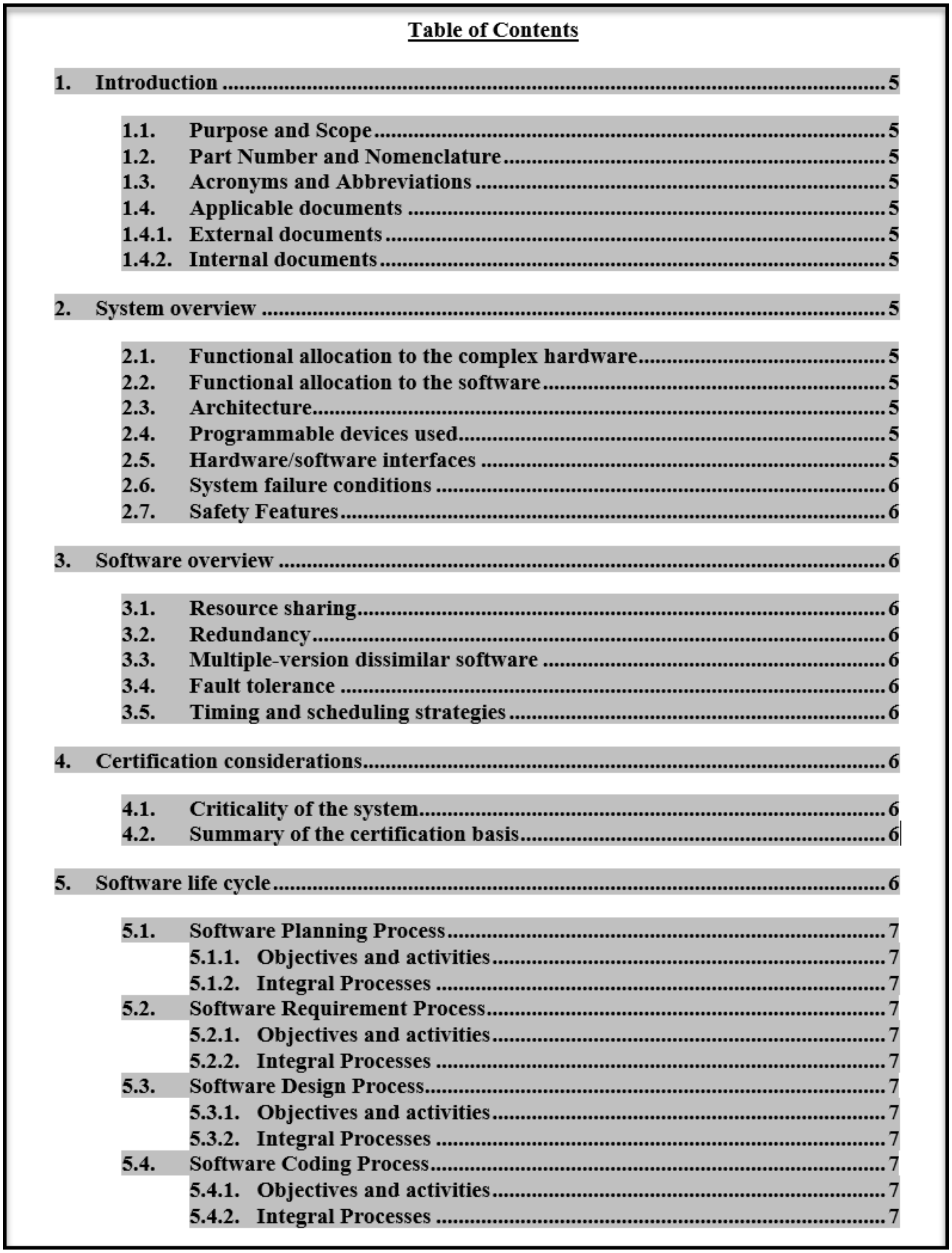


Table of Contents

Figure 15., Publicly Available PSAC Template[69]

[69] Ex. D8, Found publicly at https://www.drdo.gov.in/sites/default/files/inline-files/PSAC_template.docx. Last accessed April 23, 2023.

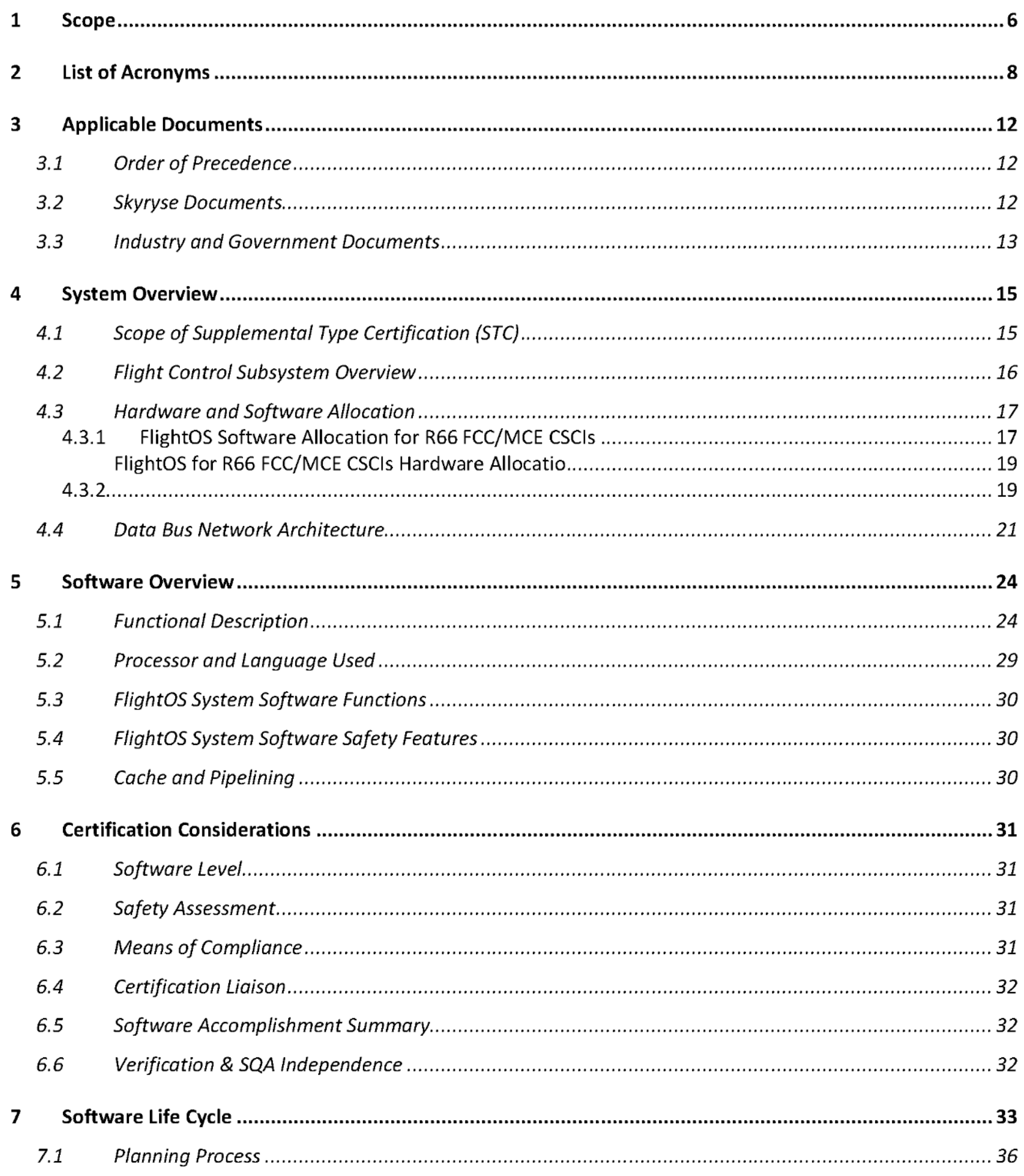


Figure 16., Crozier Decl. at Paragraph 60 – Skyryse Table of Contents

73. On pages 31 and 32, Mr. Crozier claims the PSAC of Skyryse is "[n]early identical document structure and numerous word-for word passages to Moog template document." However, as previously discussed, Mr. Crozier fails to appreciate the fact the overall structure of the PSAC follows the DO-178C

standard and as guidance is provided in DO-330 and DO-331, and the defined means of compliance is widely standardized throughout industry.[70]

74. For example, at least one earlier version of the FAA Software Approval Guidelines state that to "be consistent with prior approvals, use RTCA/DO-178B to evaluate the processes used to make the change, the changed software components, and those components affected by the software changes, using the guidelines of chapter 11 of this order and Sections 12.1.1 through 12.1.6 of RTCA/DO-178B. Affected components should be identified by performing a change impact analysis of the software changes and identifying impacts on other components, interfaces, timing, and memory (for example, control coupling analysis, data coupling analysis, timing analysis, and memory usage analysis). These analyses should also identify the level and extent of regression testing needed to verify the change."[71]

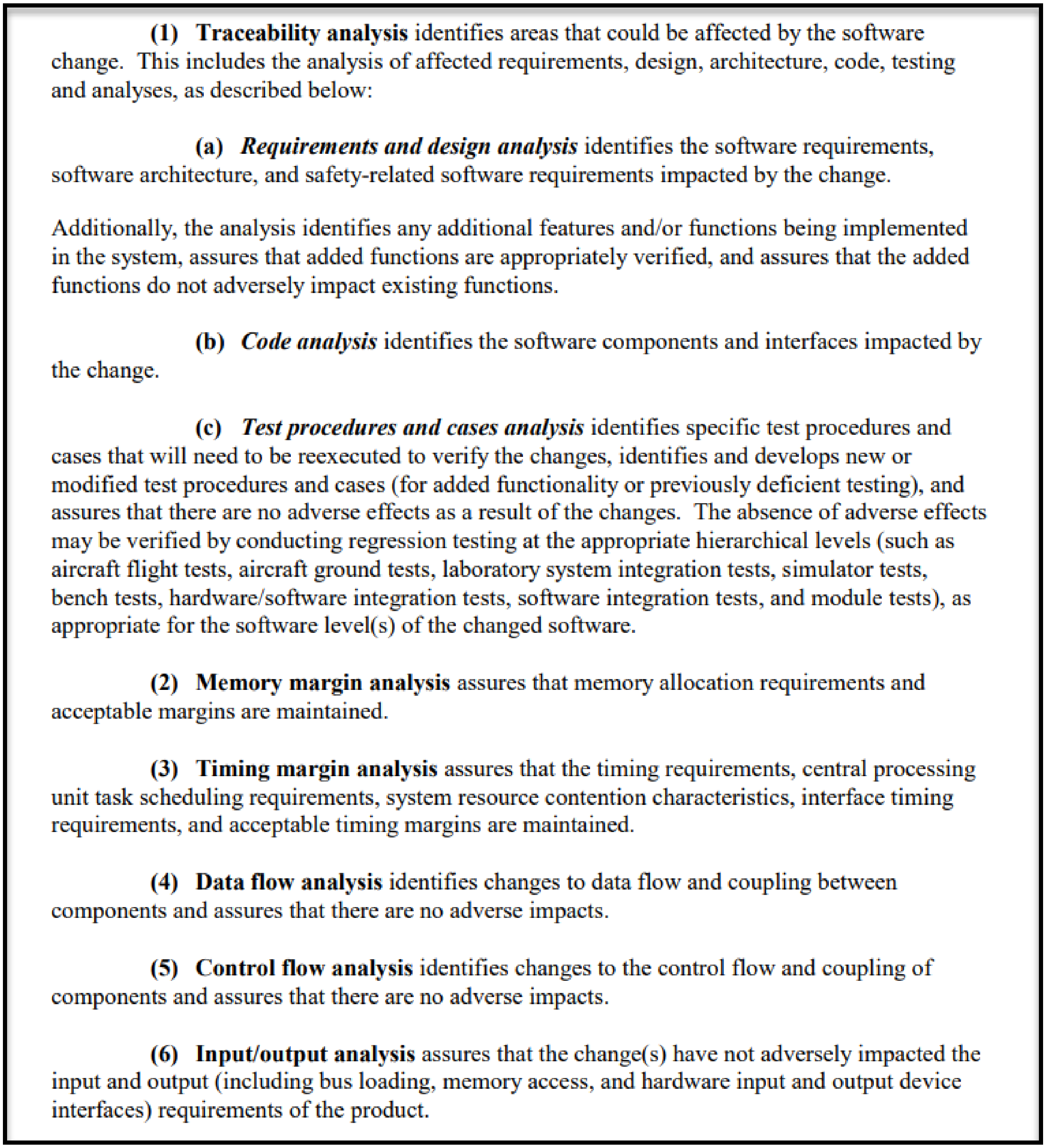
75. For example, at least one version of the FAA Software Approval Guidelines also states "[t]he applicant should identify the software changes to be incorporated in the product and perform a change impact analysis. The change impact analysis should follow a defined process to determine the potential impact of the change on continued operational safety of the aircraft. For TSO authorized equipment, the analysis should identify the intended target aircraft environment that forms the basis for the analysis."[72] The Guidelines state that "[t]his analysis also provides a basis for determining the extent of certification authority

[70] Examples provided: https://ps-engineering.com/Product_Photos/TSO/002-145-1780_PAC45T_PSAC_R1.pdf; https://cgavlas.files.wordpress.com/2018/08/psac-template.pdf; https://www.drdo.gov.in/centre_for_military_airworthines/templates, https://www.easa.europa.eu › sites › default › files › dfu › CP-ETSO template_v2020.DOC; and https://ntrs.nasa.gov/api/citations/19900011378/downloads/19900011378.pdf.).

[71] Found at https://www.faa.gov/documentLibrary/media/Order/FAA_Order_8110.49_with_chg_2.pdf. Last accessed on April 23, 2023.

[72] Found at https://www.faa.gov/documentLibrary/media/Order/FAA_Order_8110.49_with_chg_2.pdf. Last accessed on April 23, 2023.

involvement" and that the "following items should be addressed by the change impact analysis, as applicable:



(1) Traceability analysis identifies areas that could be affected by the software change. This includes the analysis of affected requirements, design, architecture, code, testing and analyses, as described below:

(a) ***Requirements and design analysis*** identifies the software requirements, software architecture, and safety-related software requirements impacted by the change.

Additionally, the analysis identifies any additional features and/or functions being implemented in the system, assures that added functions are appropriately verified, and assures that the added functions do not adversely impact existing functions.

(b) ***Code analysis*** identifies the software components and interfaces impacted by the change.

(c) ***Test procedures and cases analysis*** identifies specific test procedures and cases that will need to be reexecuted to verify the changes, identifies and develops new or modified test procedures and cases (for added functionality or previously deficient testing), and assures that there are no adverse effects as a result of the changes. The absence of adverse effects may be verified by conducting regression testing at the appropriate hierarchical levels (such as aircraft flight tests, aircraft ground tests, laboratory system integration tests, simulator tests, bench tests, hardware/software integration tests, software integration tests, and module tests), as appropriate for the software level(s) of the changed software.

(2) Memory margin analysis assures that memory allocation requirements and acceptable margins are maintained.

(3) Timing margin analysis assures that the timing requirements, central processing unit task scheduling requirements, system resource contention characteristics, interface timing requirements, and acceptable timing margins are maintained.

(4) Data flow analysis identifies changes to data flow and coupling between components and assures that there are no adverse impacts.

(5) Control flow analysis identifies changes to the control flow and coupling of components and assures that there are no adverse impacts.

(6) Input/output analysis assures that the change(s) have not adversely impacted the input and output (including bus loading, memory access, and hardware input and output device interfaces) requirements of the product.

Figure 17.,
FAA Order 8110.49–Software Approval Guidelines–with change 2[73]

[73] Found at https://www.faa.gov/documentLibrary/media/Order/FAA_Order_8110.49_with_chg_2.pdf. Last accessed on April 23, 2023.

76. In other words, Skyryse's PSAC covers the issues identified in Paragraphs 61 and 62 of Crozier's declaration because these are the very topics the FAA requires for certification purposes, which topics are publicly known and not unique to Moog.

D. Software Quality Assurance Plans (SQAP)

77. A SQAP describes the specific methods and activities that a company will employ to ensure that its software development is performed in accordance with its software development plans and PSAC. Descriptions and templates for SQAPs are readily available online from public websites. There is nothing confidential about the templates or standard categories described in those templates. Rather, SQAPs are differentiated by the specific methods, tools, and activities that individual companies use to ensure software development compliance for their specific products.

78. On pages 33 through 35, Mr. Crozier claims the SQAPs are nearly identical, except that Skyryse "removes not applicable Moog references but still retains the Moog document structure and numerous word-for-word passages and sections." Mr. Crozier again fails to appreciate that the two SQAPs are only common in structure in that both are responsive to the requirements provided in DO-178C[74], DO-330,[75] and DO-331,[76] and that the published table of contents of both companies are consistent with others in industry, and the relevant standards.[77]

79. Figure 18 below reflects a screenshot of a table of contents that Mr. Crozier has identified as Moog nonpublic information and "Evidence of Misappropriation."[78] Figure 19 reflects the DO-178C standard on which Figure 18

[74] DO-178C section 11.1.

[75] DO-330 section 11.1.

[76] DO-331 section MB 11.1.

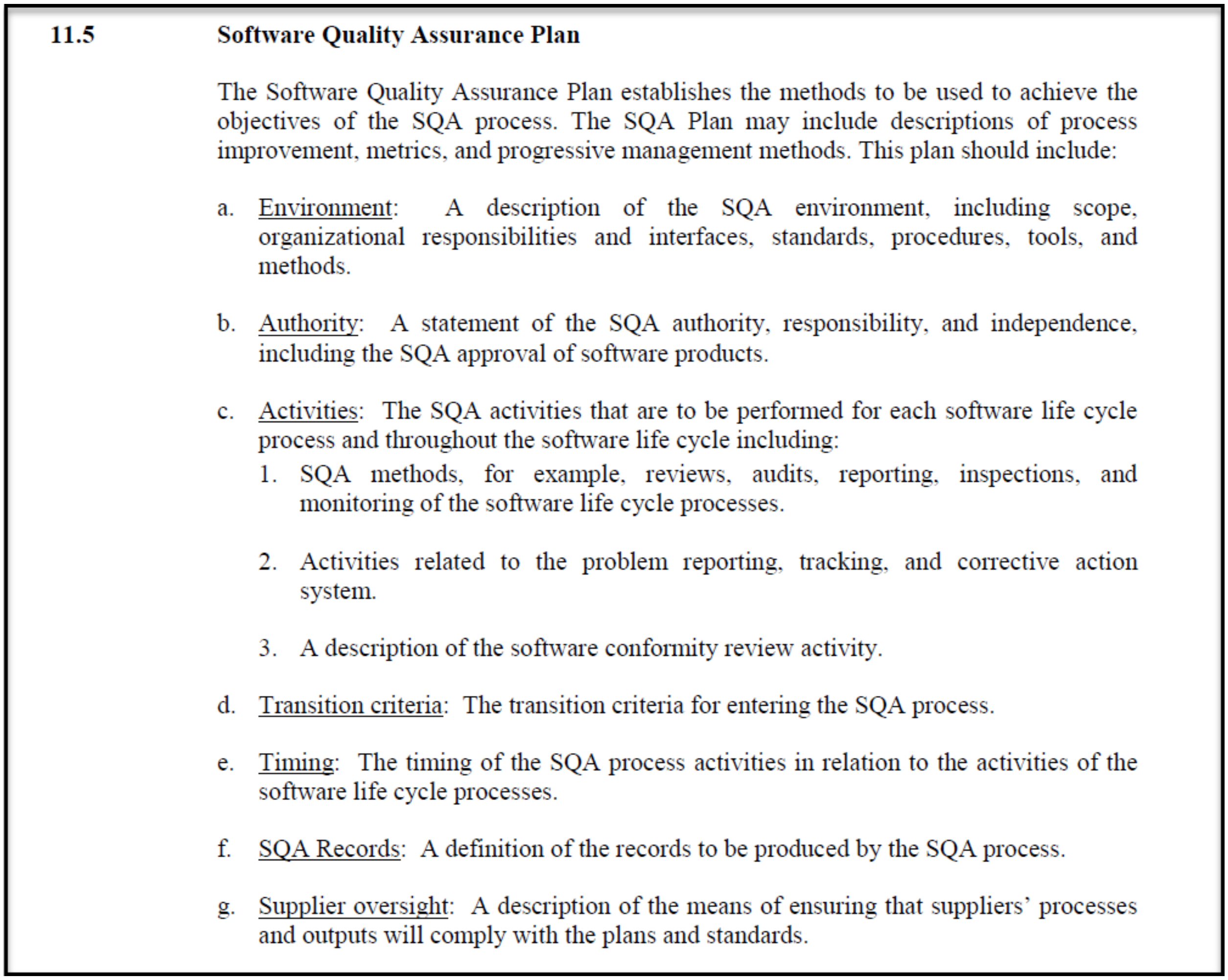
[77] https://www.renesas.cn/cn/zh/document/mat/synergy-software-quality-handbook.

[78] Crozier Decl. at 22 and ¶ 65.

is based, and Figure 20 reflects a screenshot of the table of contents from the Skyryse document identified by Mr. Crozier.

[redacted]

Figure 18., Crozier Paragraph 65 – [redacted]



11.5 Software Quality Assurance Plan

The Software Quality Assurance Plan establishes the methods to be used to achieve the objectives of the SQA process. The SQA Plan may include descriptions of process improvement, metrics, and progressive management methods. This plan should include:

a. Environment: A description of the SQA environment, including scope, organizational responsibilities and interfaces, standards, procedures, tools, and methods.

b. Authority: A statement of the SQA authority, responsibility, and independence, including the SQA approval of software products.

c. Activities: The SQA activities that are to be performed for each software life cycle process and throughout the software life cycle including:

1. SQA methods, for example, reviews, audits, reporting, inspections, and monitoring of the software life cycle processes.

2. Activities related to the problem reporting, tracking, and corrective action system.

3. A description of the software conformity review activity.

d. Transition criteria: The transition criteria for entering the SQA process.

e. Timing: The timing of the SQA process activities in relation to the activities of the software life cycle processes.

f. SQA Records: A definition of the records to be produced by the SQA process.

g. Supplier oversight: A description of the means of ensuring that suppliers' processes and outputs will comply with the plans and standards.

Figure 19.,
Publicly Available DO-178C – Section 11.5 – Software Quality Assurance Plan

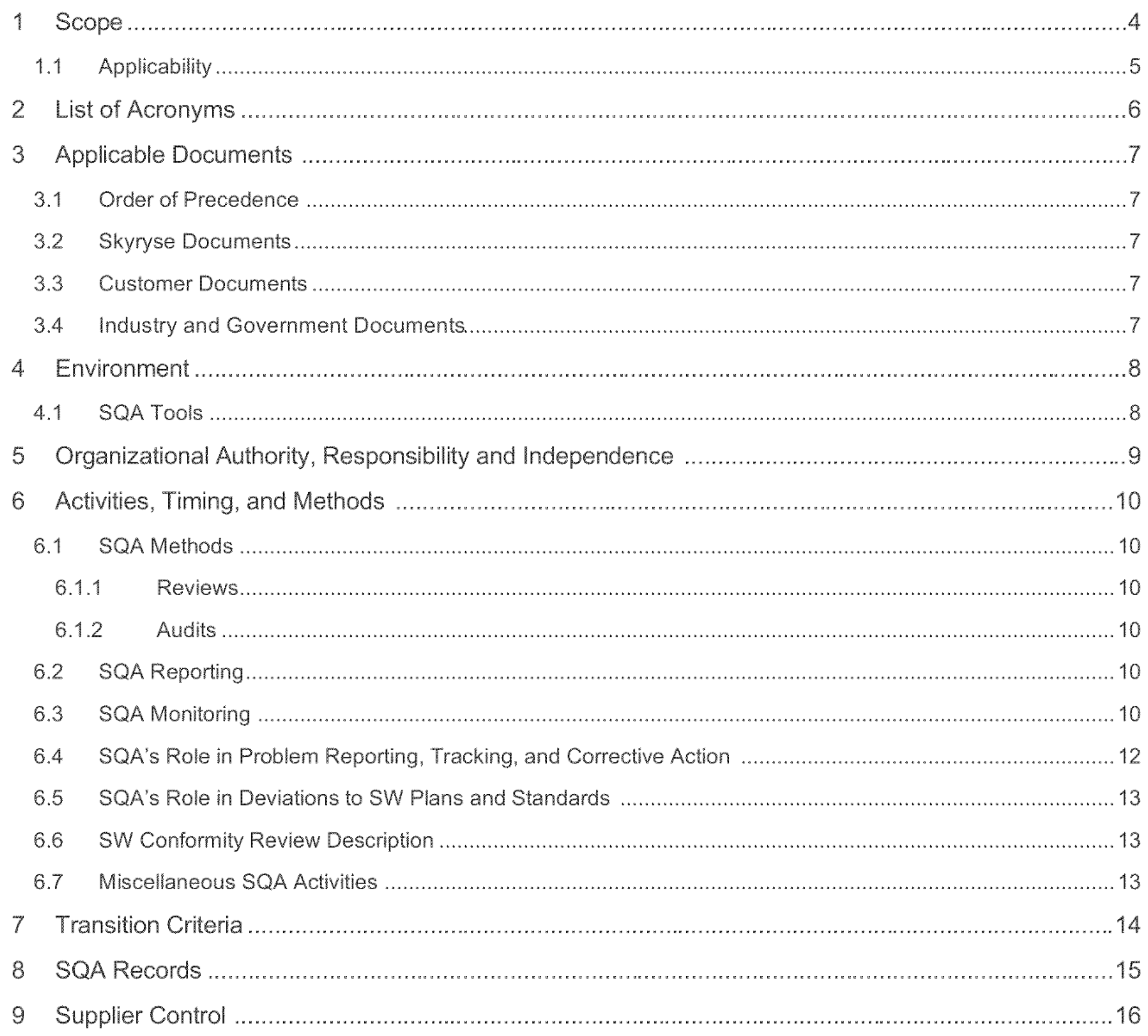


Figure 20.,
Crozier Paragraph 66 – Skyryse SKY-DOC-1019 Table of Contents

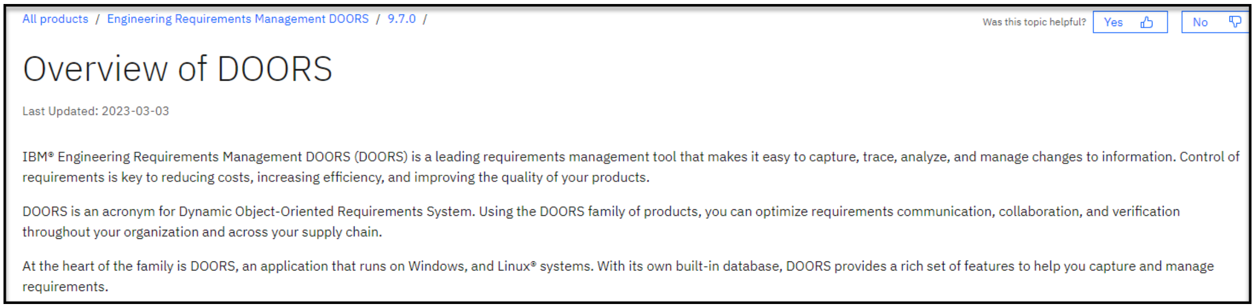
80. On pages 36 through 38, Mr. Crozier claims Skyryse used Moog's "non-public information" in crafting its SQAP. While the wording in the cited examples is similar, they are based on the requirements provided in DO-178C and consistent with examples that are publicly available and represent the standardization of the industry. For example, the Software Quality Assurance Plan Template, attached as Ex. D9, which is based on IEEE guidelines, contains sections on Reviews and Audits (Section 6) and Problem Reporting and Corrective Action (Section 8). Although this document is not based on the DO-standards, it

represents that standardization among the consensus standard organizations on structure and content for software validation and certification.

E. Software Configuration Management Plans (SCMP)

81. The SCMP describes the specific methods that a company will employ to configure its airborne software. The activities described in the SCMP are standard and must comply with DO-178C, DO-330 and DO-331. SCMP templates are readily available online. Based on my experience, there is nothing confidential about the templates or standard activities described in those templates. Rather, SCMPs are differentiated by the specific controls and configurations that an individual company employs to implement its airborne software configuration for its specific products.

82. On pages 38 and 39, Mr. Crozier claims Skyryse's reference to Dynamic Object Oriented Requirements System (DOORS) serves as an example of Skyryse taking Moog non-public information. But DOORS is widely utilized in industry and is actually embodied in the aviation industry body of knowledge as a standard process step for software certification.[79]


All products / Engineering Requirements Management DOORS / 9.7.0 /
Was this topic helpful?
Yes
No

Overview of DOORS

Last Updated: 2023-03-03

IBM® Engineering Requirements Management DOORS (DOORS) is a leading requirements management tool that makes it easy to capture, trace, analyze, and manage changes to information. Control of requirements is key to reducing costs, increasing efficiency, and improving the quality of your products.

DOORS is an acronym for Dynamic Object-Oriented Requirements System. Using the DOORS family of products, you can optimize requirements communication, collaboration, and verification throughout your organization and across your supply chain.

At the heart of the family is DOORS, an application that runs on Windows, and Linux® systems. With its own built-in database, DOORS provides a rich set of features to help you capture and manage requirements.

Figure 21., Publicly Available IBM DOORS Website[80]

83. On pages 40 through 41, Mr. Crozier submits excerpt examples of Moog's and Skyryse's SCMPs, and claims Skyryse relied on Moog's "non-public" content to create their own document. Again, Mr. Crozier fails to recognize the

[79] The Standard of Knowledge for the Aviation, Space & Defense Industry Quality Practitioner: The AS&D Quality Body of Knowledge (BoK)—Version 1, ISBN 978-1-937974-00-8, The CPK Publishing 2012;

[80] https://www.ibm.com/docs/en/elms/ermd/9.7.0?topic=overview-doors

table of contents of both documents follow the structure and requirements set forth in DO-178C[81] that are not unique to Moog. Mr. Crozier also fails to recognize that the table of contents structure that he points to as being Moog's, is widely applied in industry and is publicly available[82] and that the processes are not identical in the plans of the two companies.

84. Figure 22 below reflects a screen shot of a table of contents of a Moog SCMP that Mr. Crozier identifies as Moog nonpublic information and "Evidence of Misappropriation."[83] Figure 23 reflects a screenshot of the DO-178C standard on which Figure 22 is based. Figure 24 reflects a screenshot of a similar publicly available table of contents and Figure 25 reflects a screenshot of a Skyryse document identified by Mr. Crozier at Paragraph 75 of his declaration.

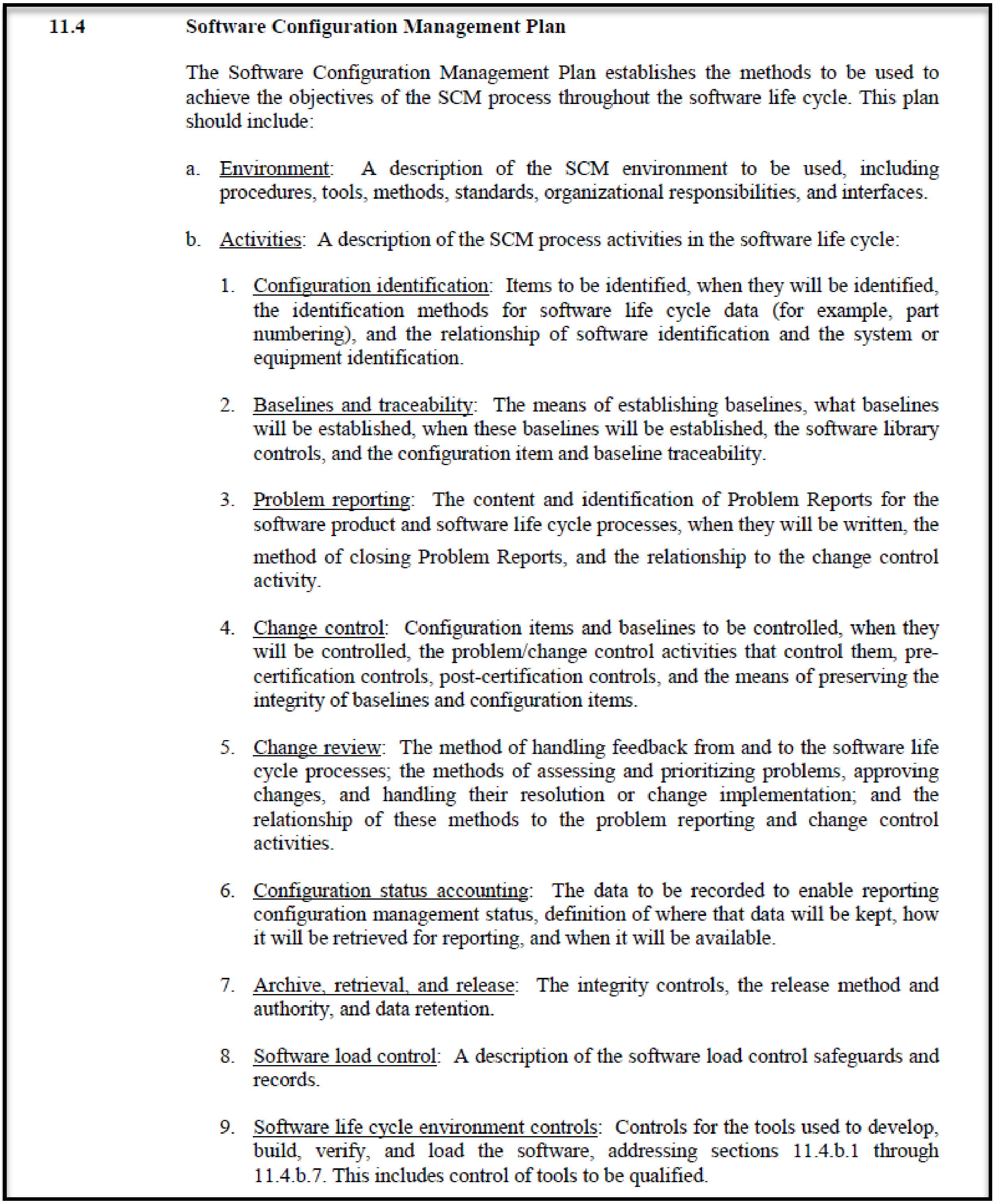
[81] DO-178C section 11.4.

[82] https://www.renesas.cn/cn/zh/document/mat/synergy-software-quality-handbook.

[83] Crozier Decl. at 22 and ¶ 74.

[redacted]

Figure 22., Crozier Paragraph 74 – [redacted]



11.4 Software Configuration Management Plan

The Software Configuration Management Plan establishes the methods to be used to achieve the objectives of the SCM process throughout the software life cycle. This plan should include:

a. Environment: A description of the SCM environment to be used, including procedures, tools, methods, standards, organizational responsibilities, and interfaces.

b. Activities: A description of the SCM process activities in the software life cycle:

1. Configuration identification: Items to be identified, when they will be identified, the identification methods for software life cycle data (for example, part numbering), and the relationship of software identification and the system or equipment identification.

2. Baselines and traceability: The means of establishing baselines, what baselines will be established, when these baselines will be established, the software library controls, and the configuration item and baseline traceability.

3. Problem reporting: The content and identification of Problem Reports for the software product and software life cycle processes, when they will be written, the method of closing Problem Reports, and the relationship to the change control activity.

4. Change control: Configuration items and baselines to be controlled, when they will be controlled, the problem/change control activities that control them, pre-certification controls, post-certification controls, and the means of preserving the integrity of baselines and configuration items.

5. Change review: The method of handling feedback from and to the software life cycle processes; the methods of assessing and prioritizing problems, approving changes, and handling their resolution or change implementation; and the relationship of these methods to the problem reporting and change control activities.

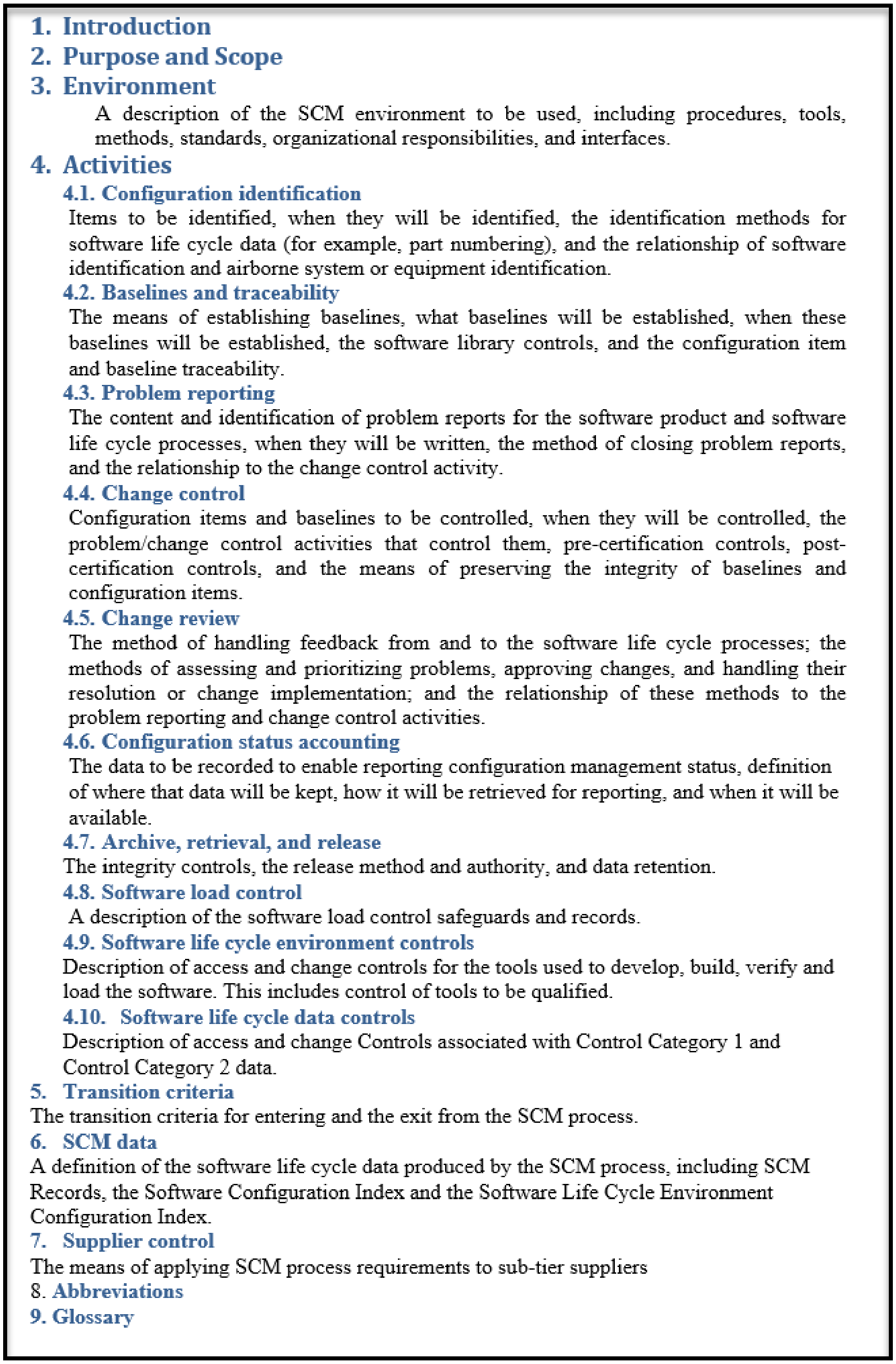
6. Configuration status accounting: The data to be recorded to enable reporting configuration management status, definition of where that data will be kept, how it will be retrieved for reporting, and when it will be available.

7. Archive, retrieval, and release: The integrity controls, the release method and authority, and data retention.

8. Software load control: A description of the software load control safeguards and records.

9. Software life cycle environment controls: Controls for the tools used to develop, build, verify, and load the software, addressing sections 11.4.b.1 through 11.4.b.7. This includes control of tools to be qualified.

Figure 23., DO-178C – Section 11.4 – Software Configuration Management Plan



1. Introduction

2. Purpose and Scope

3. Environment

A description of the SCM environment to be used, including procedures, tools, methods, standards, organizational responsibilities, and interfaces.

4. Activities

4.1. Configuration identification

Items to be identified, when they will be identified, the identification methods for software life cycle data (for example, part numbering), and the relationship of software identification and airborne system or equipment identification.

4.2. Baselines and traceability

The means of establishing baselines, what baselines will be established, when these baselines will be established, the software library controls, and the configuration item and baseline traceability.

4.3. Problem reporting

The content and identification of problem reports for the software product and software life cycle processes, when they will be written, the method of closing problem reports, and the relationship to the change control activity.

4.4. Change control

Configuration items and baselines to be controlled, when they will be controlled, the problem/change control activities that control them, pre-certification controls, post-certification controls, and the means of preserving the integrity of baselines and configuration items.

4.5. Change review

The method of handling feedback from and to the software life cycle processes; the methods of assessing and prioritizing problems, approving changes, and handling their resolution or change implementation; and the relationship of these methods to the problem reporting and change control activities.

4.6. Configuration status accounting

The data to be recorded to enable reporting configuration management status, definition of where that data will be kept, how it will be retrieved for reporting, and when it will be available.

4.7. Archive, retrieval, and release

The integrity controls, the release method and authority, and data retention.

4.8. Software load control

A description of the software load control safeguards and records.

4.9. Software life cycle environment controls

Description of access and change controls for the tools used to develop, build, verify and load the software. This includes control of tools to be qualified.

4.10. Software life cycle data controls

Description of access and change Controls associated with Control Category 1 and Control Category 2 data.

5. Transition criteria

The transition criteria for entering and the exit from the SCM process.

6. SCM data

A definition of the software life cycle data produced by the SCM process, including SCM Records, the Software Configuration Index and the Software Life Cycle Environment Configuration Index.

7. Supplier control

The means of applying SCM process requirements to sub-tier suppliers

8. Abbreviations

9. Glossary

Figure 24., Software Configuration Management Plan[84]

[84] Ex. D10, found publicly available at: https://www.drdo.gov.in/sites/default/files/inline-files/SCMP_Template_0.docx

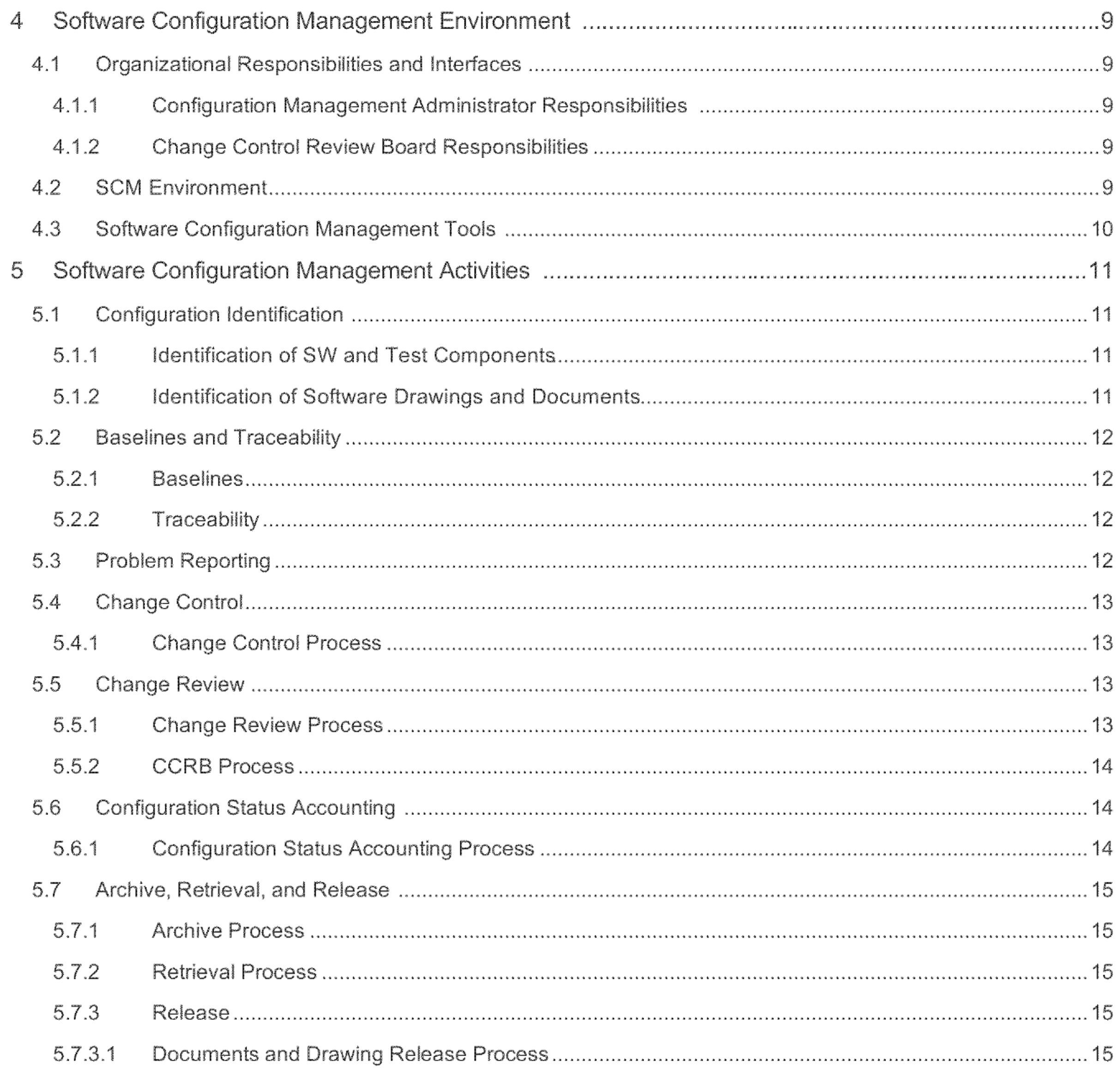


Figure 25., Crozier Paragraph 75 – Skyryse SKY-DOC-1016 Table of Contents

85. On pages 42 through 43, Mr. Crozier submits excerpt examples of Moog's and Skyryse's SCMPs, and claims Skyryse relied on Moog's "non-public" content to create their own document. Again, Mr. Crozier fails to recognize that the contents of both documents follow requirements set forth in DO-178C[85] that are not unique to Moog, or even to the aircraft industry, as reflected in the attached

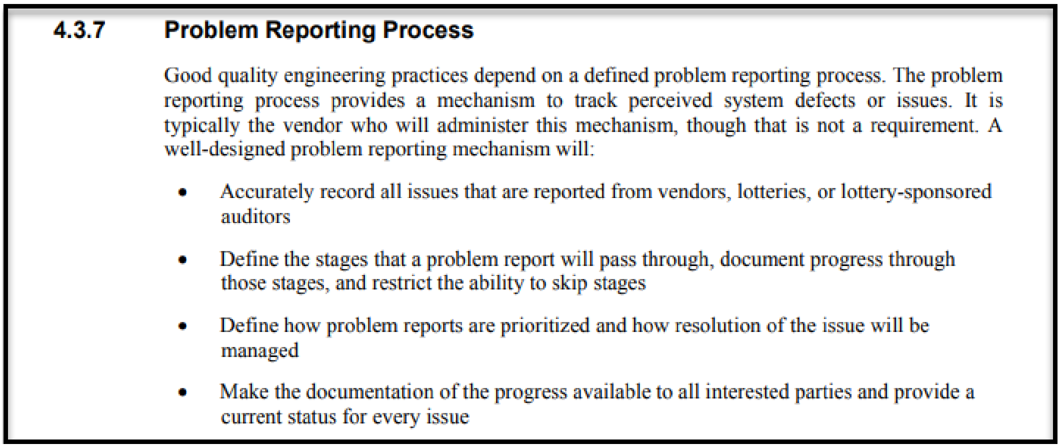
[85] DO-178C section 11.4.

Ex. D11 Quality Assurance of Product Development in the Lottery Industry: Development Process.[86]

[redacted]

Figure 26., Crozier Paragraph 76 – [redacted]

[86] https://naspl.org/img/standards/BP0402.pdf

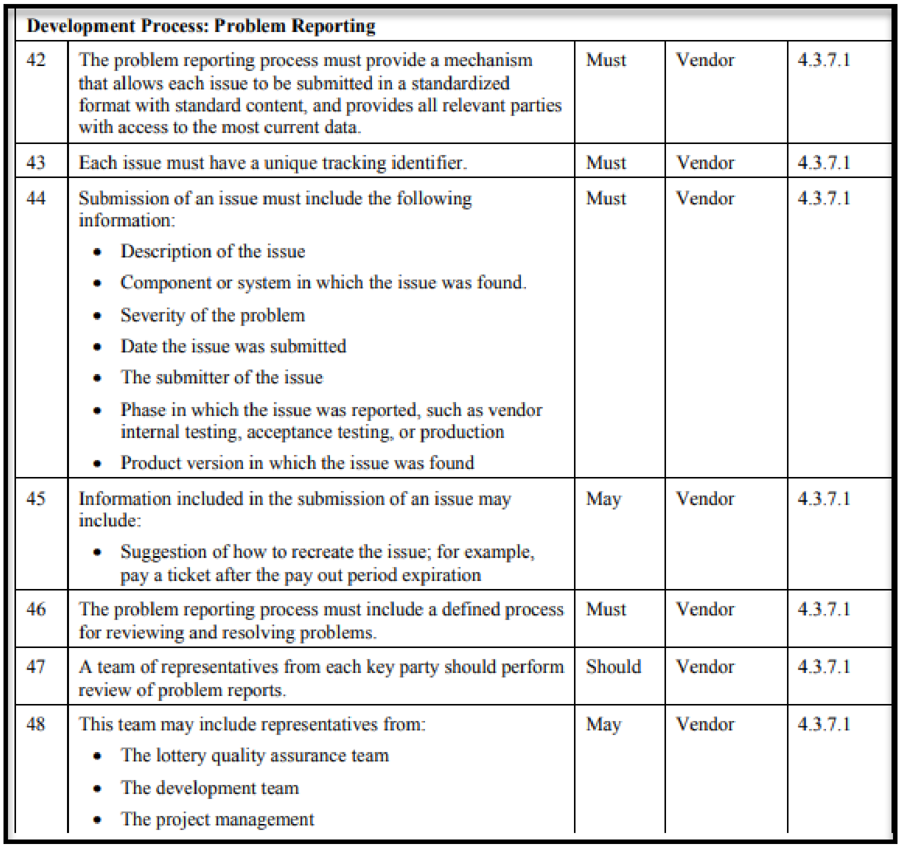


4.3.7 Problem Reporting Process

Good quality engineering practices depend on a defined problem reporting process. The problem reporting process provides a mechanism to track perceived system defects or issues. It is typically the vendor who will administer this mechanism, though that is not a requirement. A well-designed problem reporting mechanism will:

- Accurately record all issues that are reported from vendors, lotteries, or lottery-sponsored auditors
- Define the stages that a problem report will pass through, document progress through those stages, and restrict the ability to skip stages
- Define how problem reports are prioritized and how resolution of the issue will be managed
- Make the documentation of the progress available to all interested parties and provide a current status for every issue

Figure 27., Publicly Available Quality Assurance of Product Development in the Lottery Industry[87]



Development Process: Problem Reporting

42	The problem reporting process must provide a mechanism that allows each issue to be submitted in a standardized format with standard content, and provides all relevant parties with access to the most current data.	Must	Vendor	4.3.7.1
43	Each issue must have a unique tracking identifier.	Must	Vendor	4.3.7.1
44	Submission of an issue must include the following information: • Description of the issue • Component or system in which the issue was found. • Severity of the problem • Date the issue was submitted • The submitter of the issue • Phase in which the issue was reported, such as vendor internal testing, acceptance testing, or production • Product version in which the issue was found	Must	Vendor	4.3.7.1
45	Information included in the submission of an issue may include: • Suggestion of how to recreate the issue; for example, pay a ticket after the pay out period expiration	May	Vendor	4.3.7.1
46	The problem reporting process must include a defined process for reviewing and resolving problems.	Must	Vendor	4.3.7.1
47	A team of representatives from each key party should perform review of problem reports.	Should	Vendor	4.3.7.1
48	This team may include representatives from: • The lottery quality assurance team • The development team • The project management	May	Vendor	4.3.7.1

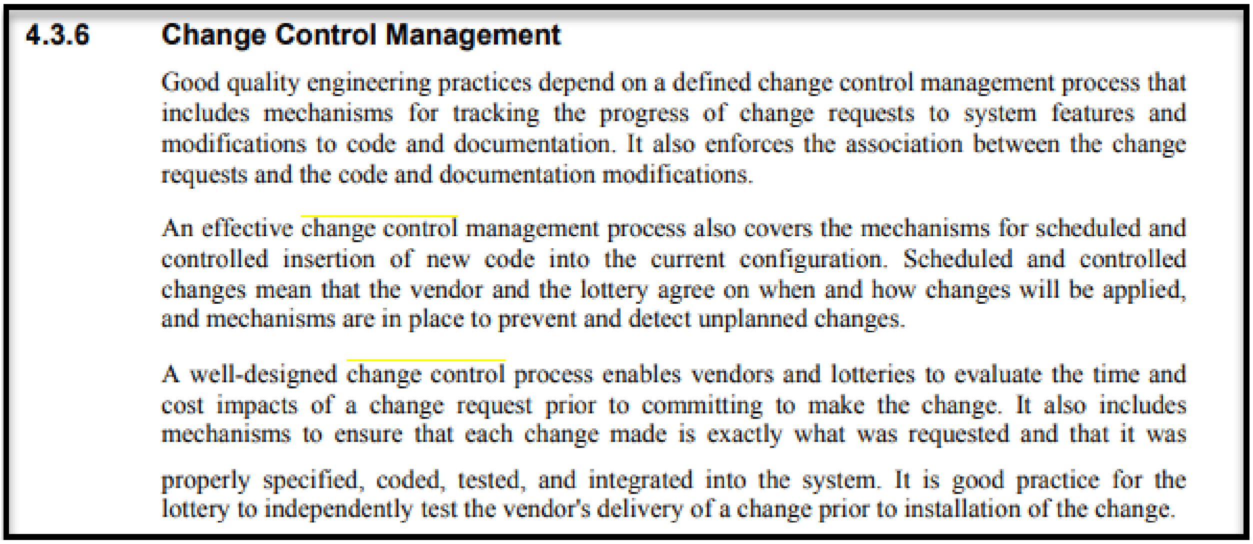
Figure 28., Publicly Available Quality Assurance of Product Development in the Lottery Industry[88]

86. I include further examples below, which are from a generic project plan template document readily found online. Although this document also is not

[87] Ex. D11, found publicly at https://naspl.org/img/standards/BP0402.pdf

[88] Ex. D11, found publicly at https://naspl.org/img/standards/BP0402.pdf

specific to the aerospace industry generally or based on the DO-standards specifically, it illustrates the basic management structure and tasks associated with ensuring that a software project meets the relevant requirements. Those same structures and tasks are consistent with the DO-standards and reflect that the same principles apply across various industries.



4.3.6 Change Control Management

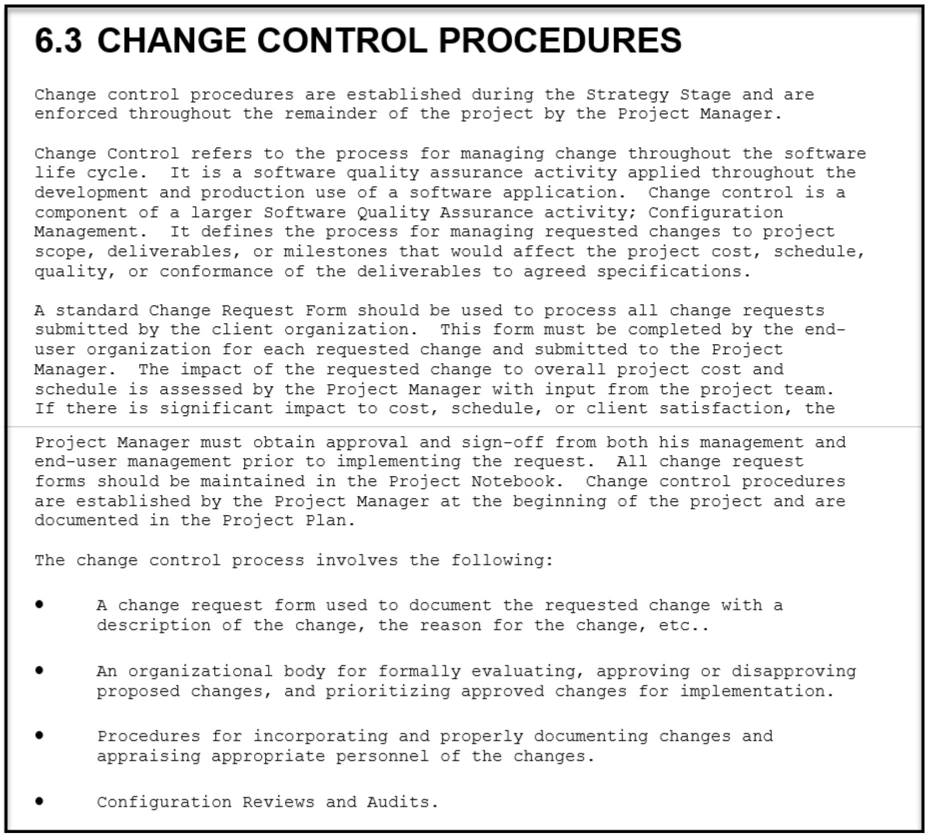
Good quality engineering practices depend on a defined change control management process that includes mechanisms for tracking the progress of change requests to system features and modifications to code and documentation. It also enforces the association between the change requests and the code and documentation modifications.

An effective change control management process also covers the mechanisms for scheduled and controlled insertion of new code into the current configuration. Scheduled and controlled changes mean that the vendor and the lottery agree on when and how changes will be applied, and mechanisms are in place to prevent and detect unplanned changes.

A well-designed change control process enables vendors and lotteries to evaluate the time and cost impacts of a change request prior to committing to make the change. It also includes mechanisms to ensure that each change made is exactly what was requested and that it was properly specified, coded, tested, and integrated into the system. It is good practice for the lottery to independently test the vendor's delivery of a change prior to installation of the change.

Figure 29., Publicly Available Quality Assurance of Product Development in the Lottery Industry[89]

[89] Ex. D11, found publicly at https://naspl.org/img/standards/BP0402.pdf



6.3 CHANGE CONTROL PROCEDURES

Change control procedures are established during the Strategy Stage and are enforced throughout the remainder of the project by the Project Manager.

Change Control refers to the process for managing change throughout the software life cycle. It is a software quality assurance activity applied throughout the development and production use of a software application. Change control is a component of a larger Software Quality Assurance activity; Configuration Management. It defines the process for managing requested changes to project scope, deliverables, or milestones that would affect the project cost, schedule, quality, or conformance of the deliverables to agreed specifications.

A standard Change Request Form should be used to process all change requests submitted by the client organization. This form must be completed by the end-user organization for each requested change and submitted to the Project Manager. The impact of the requested change to overall project cost and schedule is assessed by the Project Manager with input from the project team. If there is significant impact to cost, schedule, or client satisfaction, the Project Manager must obtain approval and sign-off from both his management and end-user management prior to implementing the request. All change request forms should be maintained in the Project Notebook. Change control procedures are established by the Project Manager at the beginning of the project and are documented in the Project Plan.

The change control process involves the following:

- A change request form used to document the requested change with a description of the change, the reason for the change, etc..
- An organizational body for formally evaluating, approving or disapproving proposed changes, and prioritizing approved changes for implementation.
- Procedures for incorporating and properly documenting changes and appraising appropriate personnel of the changes.
- Configuration Reviews and Audits.

Figure 30., – Publicly Available Project Plan Template[90]

F. Software Development Plans (SDP)

87. The SDP describes the software development process that an aviation company will follow to comply with DO-178C guidance. That guidance is standard and there are many online templates available for creating an SDP. It is the specific steps that each company describes within those templates to ensure compliance to relevant design certification requirements for a specific product, rather than the templates themselves that are unique.

88. On pages 44 through 48, Mr. Crozier shows tables of contents of both Moog's and Skyryse's respective software development plans (SDP), requirements flow illustrations, and displays two sections of the respective SDP. Mr. Crozier claims his examples show "[n]early identical document structure and numerous

[90] Ex. D12; https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Ffiles.dep.state.pa.us%2Faboutdep%2FBureau%2520of%2520Information%2520Technology%2Flib%2Finfotech%2Fsdm_documents%2Fproject_plan.doc&wdOrigin=BROWSELINK

word-for word passages to Moog template document". As previously discussed, Mr. Crozier fails to appreciate that the structure and content of the noted documents directly follow the requirements provided in DO-178C[91], DO-330,[92] and DO-331.[93]

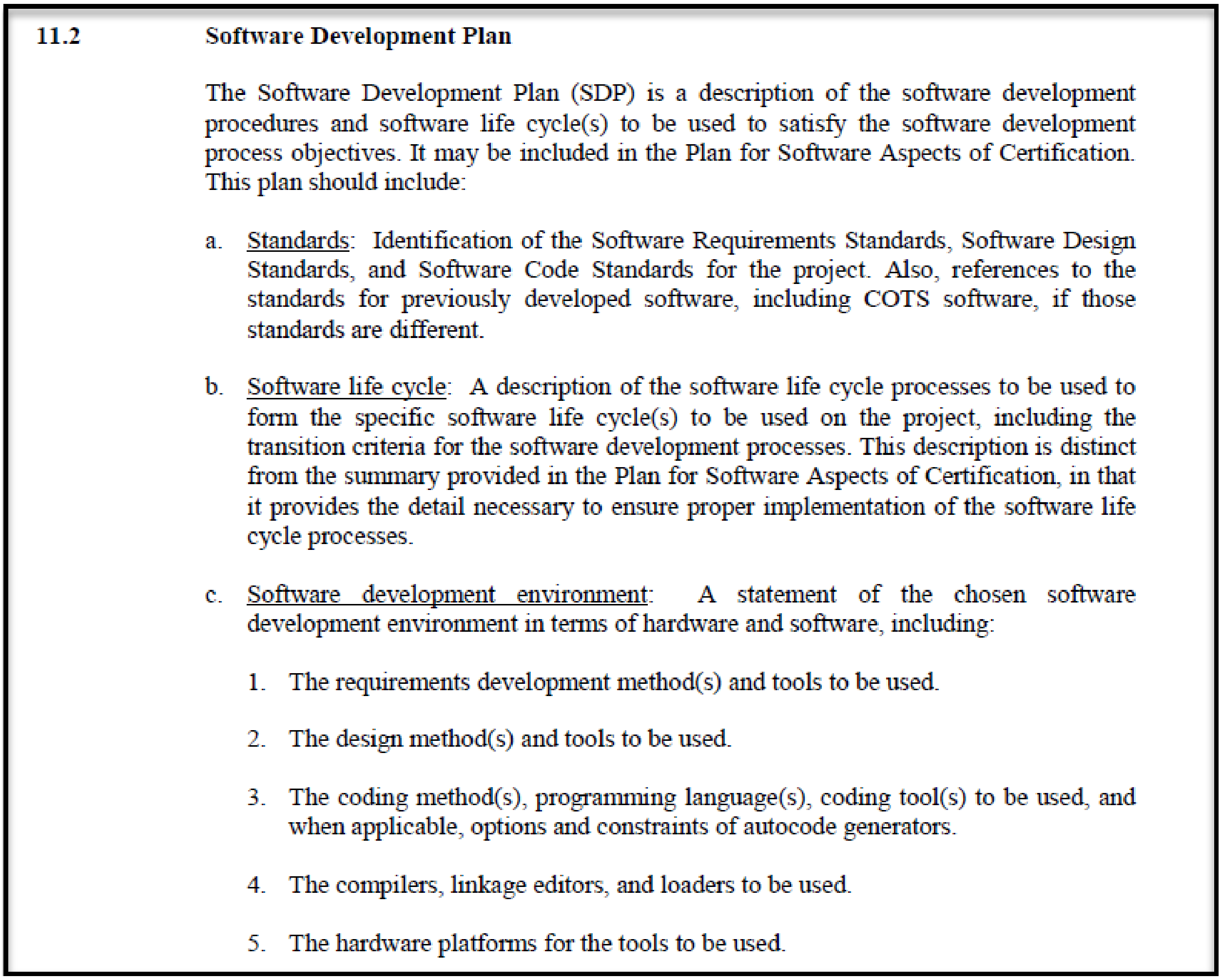
[redacted]

Figure 31., Crozier Paragraph 81 – [redacted]

[91] DO-178C section 11.2.

[92] DO-330 section 11.2.

[93] DO-331 section MB 11.2.



11.2 Software Development Plan

The Software Development Plan (SDP) is a description of the software development procedures and software life cycle(s) to be used to satisfy the software development process objectives. It may be included in the Plan for Software Aspects of Certification. This plan should include:

a. Standards: Identification of the Software Requirements Standards, Software Design Standards, and Software Code Standards for the project. Also, references to the standards for previously developed software, including COTS software, if those standards are different.

b. Software life cycle: A description of the software life cycle processes to be used to form the specific software life cycle(s) to be used on the project, including the transition criteria for the software development processes. This description is distinct from the summary provided in the Plan for Software Aspects of Certification, in that it provides the detail necessary to ensure proper implementation of the software life cycle processes.

c. Software development environment: A statement of the chosen software development environment in terms of hardware and software, including:

1. The requirements development method(s) and tools to be used.
2. The design method(s) and tools to be used.
3. The coding method(s), programming language(s), coding tool(s) to be used, and when applicable, options and constraints of autocode generators.
4. The compilers, linkage editors, and loaders to be used.
5. The hardware platforms for the tools to be used.

Figure 32., Publicly Available DO-178C – Section 11.2 – Software Development Plan

89. Moreover, the illustrations referred to as "tracing" figures are clearly provided for in DO-331, and do not appear to be of Moog origin. In order for Skyryse to comply with DO-178C, they are compelled to create a tracing figure as they have. Similar figures are easily found online through simple Google searches, as reflected below.

[redacted]

Figure 33– Crozier Paragraph 83 – [redacted]

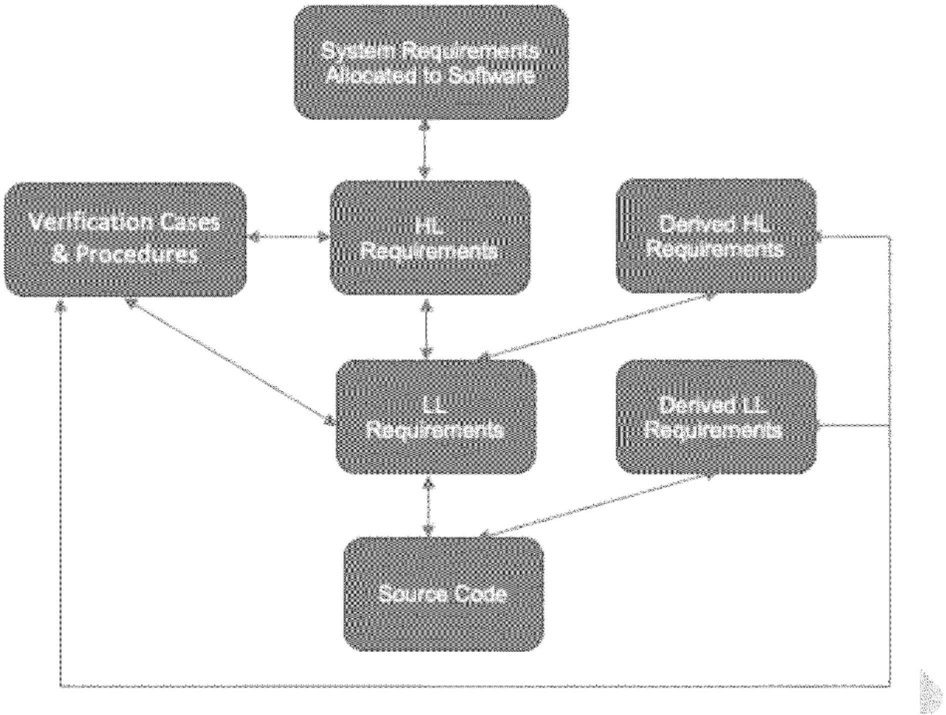
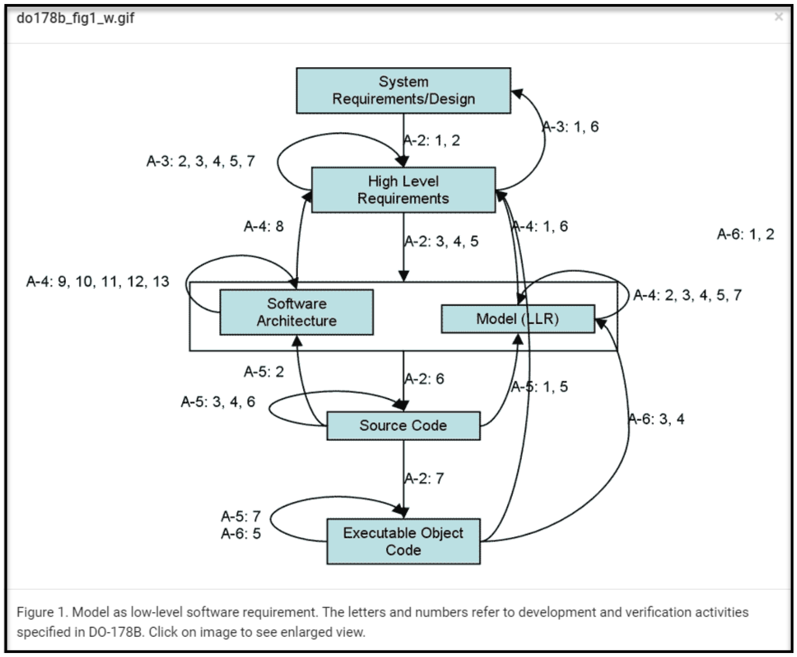


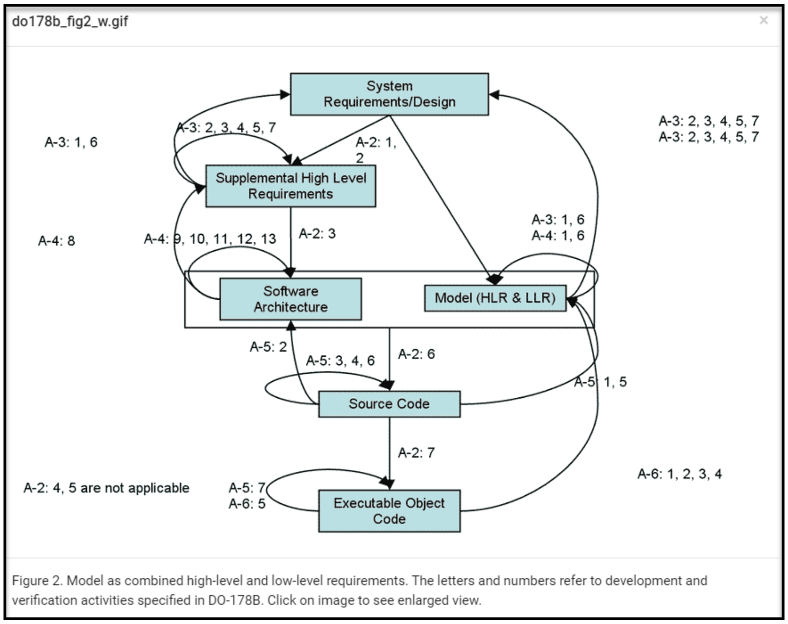
Figure 34 – Crozier Paragraph 84 – Skyryse SDP Template Tracing Figure



do178b_fig1_w.gif

Figure 1. Model as low-level software requirement. The letters and numbers refer to development and verification activities specified in DO-178B. Click on image to see enlarged view.

Figure 35., Publicly Available Model Based Design for DO-178B[94]



do178b_fig2_w.gif

Figure 2. Model as combined high-level and low-level requirements. The letters and numbers refer to development and verification activities specified in DO-178B. Click on image to see enlarged view.

Figure 36., Publicly Available Model Based Design for DO-178B[95]

[94] Ex. D13 found at https://www.mathworks.com/company/newsletters/articles/model-based-design-for-do-178b.html; last accessed on April 23, 2023.

[95] Ex. D13 found at https://www.mathworks.com/company/newsletters/articles/model-based-design-for-do-178b.html; last accessed on April 23, 2023.

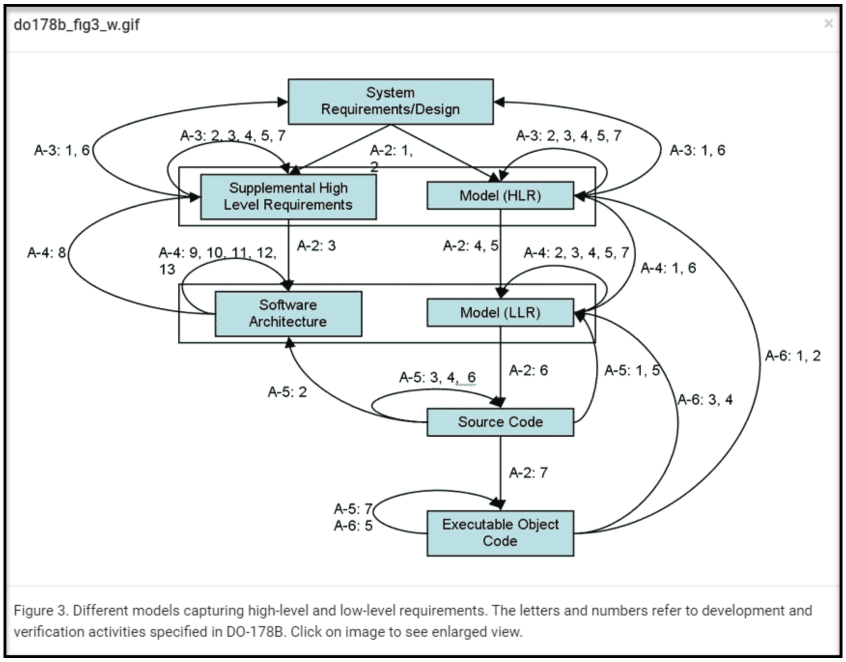


Figure 3. Different models capturing high-level and low-level requirements. The letters and numbers refer to development and verification activities specified in DO-178B. Click on image to see enlarged view.

Figure 37., Publicly Available Model Based Design for DO-178B[96]

90. The two process examples presented by Mr. Crozier (pages 47 and 48) are not identical, and each uniquely complies with the standard requirements of DO-178C. Any similarity between these examples appears to be based on the commonality of the DO-178C standard, from which both documents are derived.

91. On pages 49 and 50 of Mr. Crozier's declaration, he claims Skyryse's Software Configuration Management Plan, Software Development Plan, and Software Quality Assurance Plan all incorporated "Moog processes completely." As previously addressed in this declaration, Skyryse's procedures and planning documents appear to be common to Moog's in the respect they both comply with the industry standard DO-178C, DO-330, and DO-331 and both benefit from the highly standardized reality of the aerospace industry.

G. JIRA Problem Report Processing

92. On pages 11 and 21 of Mr. Crozier's declaration, he makes it sound as if "JIRA" is proprietary and/or unique to Moog. JIRA is a commercially available

[96] Ex. D13 found at https://www.mathworks.com/company/newsletters/articles/model-based-design-for-do-178b.html; last accessed on April 23, 2023.

project tracking software, with available free download, that is available for any person to use.[97] Atlassian, the developer of JIRA also offers many downloadable templates to manage unique software projects.[98]

93. Mr. Crozier's claim that the Skyryse Problem Report Process Using JIRA has a "nearly identical structure" and utilizes "identical word-for-word passages"[99] fails to recognize that JIRA is a commercially available software and that the roles and responsibilities of technical review boards are standardized within industry. For example, a Technical Review Board is the common term applied to the decision-making process used by a "a team of qualified personnel ... examines the suitability of the software product for its intended use and identifies discrepancies from specifications and standards. Technical reviews may also provide recommendations of alternatives and examination of various alternatives."[100]

94. As another example, Mr. Crozier points to Diagrams in Paragraphs 55 and 56 of his declaration, which he claims one of "just a few examples of copying from the Moog document" by Skyryse.[101] But the image he points to as "Evidence of Misappropriation"[102] is a publicly available and easily accessible through Atlassian.

[97] https://www.atlassian.com/software/jira.

[98] https://www.atlassian.com/software/jira/templates.

[99] Crozier Decl. pages 22 through 26.

[100] IEEE Std. 1028-1997, *IEEE Standard for Software Reviews*, clause 3.7.

[101] Crozier Decl. ¶ 48

[102] Crozier Decl. at 22.

[redacted]

Figure 38., Crozier Decl. at Paragraph 55, [redacted]

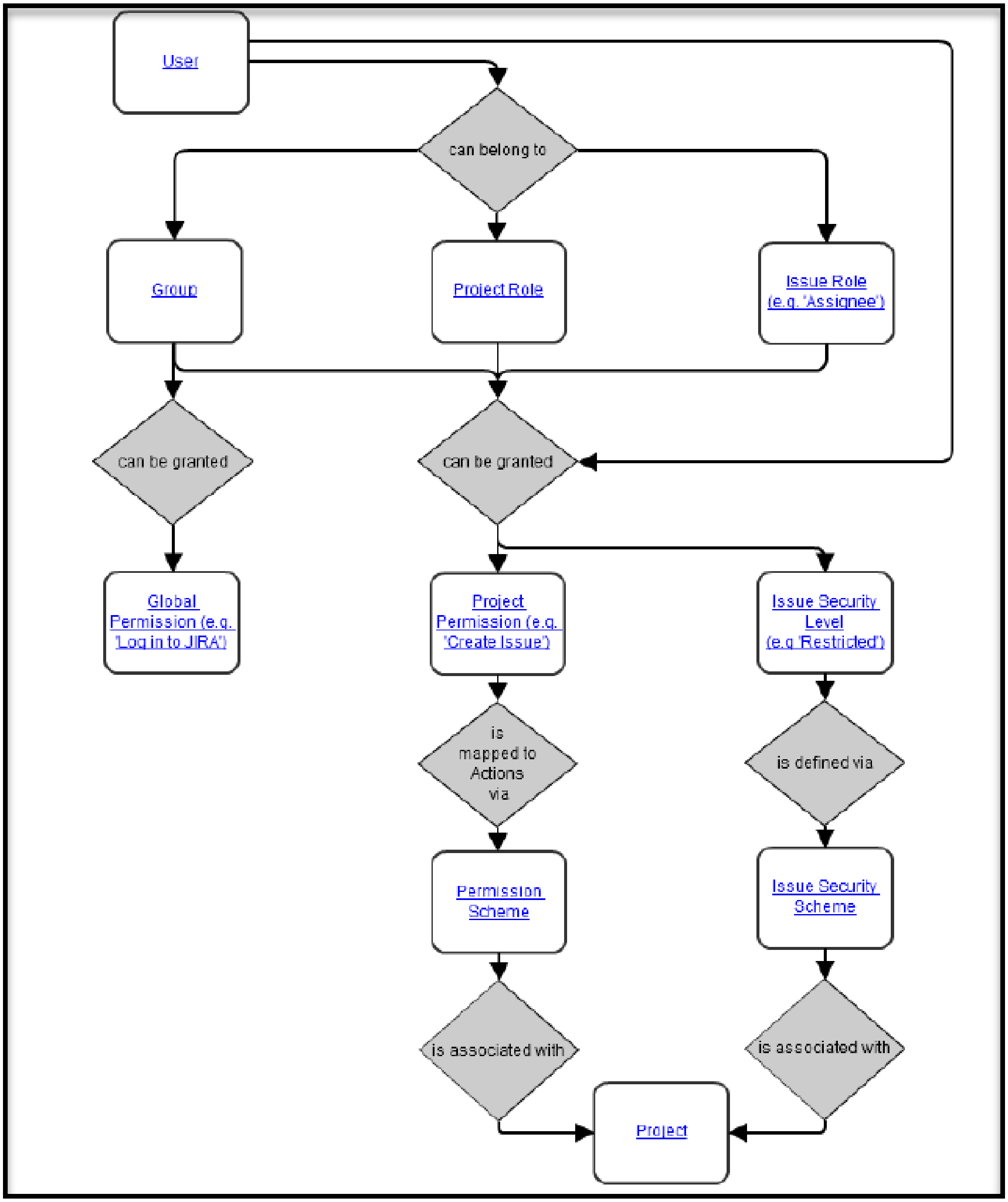


Figure 39., Identical figure from publicly available Atlassian website[103]

95. The definitions contained in the documents are also derived from industry standards. The problem report processes that Mr. Crozier discusses on pages 25 to 27 of his disclosure simply illustrates the industry standardization of

[103] Ex. D14, found publicly at https://confluence.atlassian.com/adminjiraserver0820/configuring-permissions-1095776915.html

the process for processing problem reports as provided for in DO-178C, DO-330, DO-331, FAA Order 8110.49, and FAA AC 20-115D.[104]

H. Real Time Operating Systems (RTOS)

96. On pages 58 through 61 of his declaration, Mr. Crozier claims Skyryse's "SRTOS operating system is copied directly from the Moog eRTOS operating system." But Mr. Crozier did not compare the source code for SRTOS and eRTOS,[105] which in my experience would be necessary to support a conclusion of copying. He also claims Skyryse's "SRTOS html files which are eventually compiled to the .chm file (.chm file is a compress html file) shows that it contains numerous identical or slightly modified figures (i.e.: SRTOS replaces eRTOS), identical document structure and [a] number word-for-word passages to Moog eRTOS.chm files." What he does not discuss, and as addressed in the Background section of this declaration, is how many variants there are of Real-Time Operation Systems (RTOS), many of which are open-source and free for use, and the extent to which SRTROS or eRTOS may be based on such open-source RTOSs, which could explain commonality if it exists.[106] It must also be understood that RTOS testing is also not unique to Moog and the requirements for such are grounded in DO-178; and that as the software utilized for testing is open-source or commercially available, it is not proprietary to Moog.

[104] The following are examples of publicly available standardized process of problem reporting and technical review boards: https://www.cms.gov/tra/Content/Foundation/FD_0060_Foundation_TRB.htm; https://www.dau.edu/tools/se-brainbook/Pages/Technical_Reviews_and_Audits.aspx; https://en.wikipedia.org/wiki/Software_technical_review; https://confluence.atlassian.com/jirakb/reporting-in-jira-461504615.htmL.

[105] Crozier Dep. 113:23-114:1, 115:14-19.

[106] https://en.wikipedia.org/wiki/Comparison_of_real-time_operating_systems); *see also* Crozier Dep. 109:7-110:3 (Crozier confirming that "RTOS aren't' something that's unique to Moog" and "are available for purchase from third parties" and also available as "open source.").

V. RESERVATION OF RIGHTS

97. I reserve the right to supplement and/or revise this and other reports and further specifically reserve the right to rebut any opinions provided by opposing experts.

I declare, under penalty of perjury, that the foregoing is true and correct to the best of my knowledge.

Executed on 24 April 2023, in Vieques, Puerto Rico.

Michael J. Dreikorn, Ed.D.